

# Package ‘GenVisR’

December 6, 2024

**Title** Genomic Visualizations in R

**Version** 1.38.0

**Maintainer** Zachary Skidmore <zlskidmore@gmail.com>

**Description** Produce highly customizable publication quality graphics for genomic data primarily at the cohort level.

**Depends** R (>= 3.3.0), methods

**Imports** AnnotationDbi, biomaRt (>= 2.45.8), BiocGenerics, Biostrings, DBI, GenomicFeatures, GenomicRanges (>= 1.25.4), ggplot2 (>= 2.1.0), gridExtra (>= 2.0.0), gtable, gtools, IRanges (>= 2.7.5), plyr (>= 1.8.3), reshape2, Rsamtools, scales, viridis, data.table, BSgenome, GenomeInfoDb, VariantAnnotation

**License** GPL-3 + file LICENSE

**BugReports** <https://github.com/griffithlab/GenVisR/issues>

**biocViews** Infrastructure, DataRepresentation, Classification, DNASEq

**LazyData** true

**Suggests** BiocStyle, BSgenome.Hsapiens.UCSC.hg19, knitr, RMySQL, roxygen2, testthat, TxDb.Hsapiens.UCSC.hg19.knownGene, rmarkdown, vdiff, formatR, TxDb.Hsapiens.UCSC.hg38.knownGene, BSgenome.Hsapiens.UCSC.hg38

**VignetteBuilder** knitr

**RoxygenNote** 7.2.3

**Encoding** UTF-8

**NeedsCompilation** no

**Author** Zachary Skidmore [aut, cre], Alex Wagner [aut], Robert Lesurf [aut], Katie Campbell [aut], Jason Kunisaki [aut], Obi Griffith [aut], Malachi Griffith [aut]

**Collate** 'AllGenerics.R' 'Clinical-class.R' 'GMS\_Virtual-class.R' 'GMS-class.R' 'GMS\_v4-class.R' 'GenVisR.R' 'Lollipop-class.R' 'MutSpectra-class.R' 'MutationAnnotationFormat\_Virtual-class.R' 'MutationAnnotationFormat-class.R' 'MutationAnnotationFormat\_v1.0-class.R' 'MutationAnnotationFormat\_v2.0-class.R' 'MutationAnnotationFormat\_v2.1-class.R' 'MutationAnnotationFormat\_v2.2-class.R' 'MutationAnnotationFormat\_v2.3-class.R'

'MutationAnnotationFormat\_v2.4-class.R' 'Rainfall-class.R'  
 'VEP\_Virtual-class.R' 'VEP-class.R' 'VEP\_v88-class.R'  
 'Waterfall-class.R' 'cnFreq.R' 'cnFreq\_buildMain.R'  
 'cnFreq\_disjoin.R' 'cnFreq\_qual.R' 'cnSpec.R'  
 'cnSpec\_buildMain.R' 'cnSpec\_qual.R' 'cnView.R'  
 'cnView\_buildMain.R' 'cnView\_qual.R' 'compIdent.R'  
 'compIdent\_bamRcnt.R' 'compIdent\_bamRcnt\_qual.R'  
 'compIdent\_buildMain.R' 'compIdent\_format.R' 'covBars.R'  
 'covBars\_buildMain.R' 'covBars\_qual.R' 'deprecated.R'  
 'genCov.R' 'genCov\_alignPlot.R'  
 'genCov\_assign\_ggplotGrob\_height.R'  
 'genCov\_assign\_ggplotGrob\_width.R' 'genCov\_buildCov.R'  
 'genCov\_buildTrack.R' 'genCov\_extr\_ggplotGrob\_height.R'  
 'genCov\_extr\_ggplotGrob\_width.R' 'genCov\_qual.R'  
 'genCov\_trackViz.R' 'geneViz.R' 'geneViz\_Granges2dataframe.R'  
 'geneViz\_buildGene.R' 'geneViz\_calcGC.R'  
 'geneViz\_cdsFromTXID.R' 'geneViz\_extrCDS.R' 'geneViz\_extrUTR.R'  
 'geneViz\_formatUTR.R' 'geneViz\_formatcds.R'  
 'geneViz\_mapCoordSpace.R' 'geneViz\_mapCovCoordSpace.R'  
 'geneViz\_mergeRegions.R' 'geneViz\_mergeTypeRegions.R'  
 'geneViz\_mergeTypes.R' 'ideoView.R' 'ideoView\_buildMain.R'  
 'ideoView\_formatCytobands.R' 'ideoView\_qual.R' 'lohSpec.R'  
 'lohSpec\_buildMain.R' 'lohSpec\_fileGlob.R' 'lohSpec\_lohCalc.R'  
 'lohSpec\_qual.R' 'lohSpec\_slidingWindow.R' 'lohSpec\_stepCalc.R'  
 'lohSpec\_tileCalc.R' 'lohSpec\_tilePosition.R'  
 'lohSpec\_tileWindow.R' 'lohSpec\_windowPosition.R' 'lohView.R'  
 'lohView\_buildMain.R' 'lohView\_qual.R' 'multi\_align.R'  
 'multi\_buildClin.R' 'multi\_chrBound.R' 'multi\_cytobandRet.R'  
 'multi\_selectOut.R' 'multi\_subsetChr.R'

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

**git\_branch** RELEASE\_3\_20

**git\_last\_commit** 84e2b21

**git\_last\_commit\_date** 2024-10-29

**Repository** Bioconductor 3.20

**Date/Publication** 2024-12-05

## Contents

brcaMAF . . . . .	4
Clinical-class . . . . .	4
cnFreq . . . . .	5
cnSpec . . . . .	7
cnView . . . . .	8
compIdent . . . . .	10
covBars . . . . .	11
cytoGeno . . . . .	12
drawPlot . . . . .	12
genCov . . . . .	13
geneViz . . . . .	15
GenVisR . . . . .	17

getData	17
getDescription	19
getGrob	19
getHeader	20
getMeta	20
getMutation	21
getPath	22
getPosition	22
getSample	23
getVersion	24
GMS-class	24
GMS_v4-class	25
GMS_Virtual-class	26
HCC1395_Germline	26
HCC1395_N	27
HCC1395_T	27
hg19chr	28
ideoView	28
lohSpec	29
lohView	31
lollipop	32
Lollipop-class	32
LucCNseg	34
MutationAnnotationFormat-class	34
MutationAnnotationFormat_v1.0-class	35
MutationAnnotationFormat_v2.0-class	36
MutationAnnotationFormat_v2.1-class	36
MutationAnnotationFormat_v2.2-class	37
MutationAnnotationFormat_v2.3-class	37
MutationAnnotationFormat_v2.4-class	38
MutationAnnotationFormat_Virtual-class	39
MutSpectra-class	39
PIK3CA	40
Rainfall-class	41
SNPloci	42
TvTi	42
VEP-class	44
VEP_v88-class	45
VEP_Virtual-class	45
waterfall	46
Waterfall-class	46
writeData	49

---

brcaMAF	<i>Truncated BRCA MAF file</i>
---------	--------------------------------

---

**Description**

A data set containing 50 samples corresponding to "Breast invasive carcinoma" originating from the TCGA project in .maf format (version 2.4): <https://wiki.nci.nih.gov/display/TCGA/TCGA+MAF+Files#TCGAMAFFile>  
 BRCA:Breastinvasivecarcinoma, /dccfiles\_prod/tcgafiles/distro\_ftpusers/anonymous/tumor/brca/gsc/genome.wustl.edu/

**Usage**

```
data(brcaMAF)
```

**Format**

a data frame with 2773 observations and 55 variables

**Value**

Object of class data drame

---

Clinical-class	<i>Class Clinical</i>
----------------	-----------------------

---

**Description**

An S4 class to store clinical information and plots, under development!!!

**Usage**

```
Clinical(  
  path,  
  inputData = NULL,  
  inputFormat = c("wide", "long"),  
  legendColumns = 1,  
  palette = NULL,  
  clinicalLayers = NULL,  
  verbose = FALSE  
)
```

**Arguments**

path	String specifying the path to clinical data, file must have the column "sample".
inputData	Optional data.table or data.frame object holding clinical data, used only if path is not specified. Data must have the column "sample".
inputFormat	String specifying the input format of the data given, one of wide or long format (see details).
legendColumns	Integer specifying the number of columns in the legend.
palette	Named character vector supplying colors for clinical variables.
clinicalLayers	list of ggplot2 layers to be passed to the plot.
verbose	Boolean specifying if progress should be reported.

## Details

The `Clinical()` function is a constructor to create a `GenVisR` object of class `Clinical`. This is used to both display clinical data in the form of a heatmap and to add clinical data to various `GenVisR` plots. Input to this function can be either the path to a file containing clinical information using the parameter `"path"`, or alternatively a `data.table` object if this information into R. By default the input is assumed to be in a wide format where each variable has it's own column, in such cases the data will be coerced into a long format where there is a key->value pair mapping to the data. The assumption of "wide"/"long" format can be changed with the `"inputFormat"` parameter, in both cases there should be a column called `"sample"` within the data supplied which is used as an id variable.

## Slots

`clinicalGrob` `gtable` object for the clinical plot.

`clinicalLayers` list of `ggtheme` or `ggproto` objects used to build the plot.

`clinicalData` `data.table` object to store clinical data

## See Also

[getData](#)

[drawPlot](#)

---

cnFreq

*Construct copy-number frequency plot*

---

## Description

Given a data frame construct a plot to display copy number changes across the genome for a group of samples.

## Usage

```
cnFreq(  
  x,  
  CN_low_cutoff = 1.5,  
  CN_high_cutoff = 2.5,  
  plot_title = NULL,  
  CN_Loss_colour = "#002EB8",  
  CN_Gain_colour = "#A30000",  
  x_title_size = 12,  
  y_title_size = 12,  
  facet_lab_size = 10,  
  plotLayer = NULL,  
  plotType = "proportion",  
  genome = "hg19",  
  plotChr = NULL,  
  out = "plot"  
)
```

**Arguments**

x	Object of class data frame with rows representing genomic segments. The data frame must contain columns with the following names "chromosome", "start", "end", "segmean", and "sample". Coordinates should be 1-based space.
CN_low_cutoff	Numeric value representing the point at or below which copy number alterations are considered losses. Only used if x represents CN values.
CN_high_cutoff	Numeric value representing the point at or above which copy number alterations are considered gains. Only used if x represents CN values.
plot_title	Character string specifying the title to display on the plot.
CN_Loss_colour	Character string specifying the colour value for copy number losses.
CN_Gain_colour	Character string specifying the colour value for copy number gains.
x_title_size	Integer specifying the size of the x-axis title.
y_title_size	Integer specifying the size of the y-axis title.
facet_lab_size	Integer specifying the size of the faceted labels plotted.
plotLayer	Valid ggplot2 layer to be added to the plot.
plotType	Character string specifying the type of values to plot. One of "proportion" or "frequency"
genome	Character string specifying a valid UCSC genome (see details).
plotChr	Character vector specifying specific chromosomes to plot, if NULL all chromosomes for the genome selected are displayed.
out	Character vector specifying the the object to output, one of "data", "grob", or "plot", defaults to "plot" (see returns).

**Details**

cnFreq requires the location of chromosome boundaries for a given genome assembly in order to ensure the entire chromosome space is plotted. As a convenience this information is available to cnSpec for the following genomes "hg19", "hg38", "mm9", "mm10", "rn5" and can be retrieved by supplying one of the afore mentioned assemblies via the 'genome' parameter. If a genome assembly is supplied to the 'genome' parameter and is unrecognized cnSpec will attempt to query the UCSC MySQL database for the required information. If genomic segments are not identical across all samples the algorithm will attempt to perform a disjoint operation splitting existing segments such that there are no overlaps. The 'plotLayer' parameter can be used to add an additional layer to the ggplot2 graphic (see vignette).

**Value**

One of the following, a dataframe containing data to be plotted, a grob object, or a plot.

**Examples**

```
# plot on internal GenVisR dataset
cnFreq(LucCNseg)
```

---

`cnSpec`*Construct copy-number cohort plot*

---

## Description

Given a data frame construct a plot to display copy-number calls for a cohort of samples.

## Usage

```
cnSpec(  
  x,  
  y = NULL,  
  genome = "hg19",  
  plot_title = NULL,  
  CN_Loss_colour = "#002EB8",  
  CN_Gain_colour = "#A30000",  
  x_title_size = 12,  
  y_title_size = 12,  
  facet_lab_size = 10,  
  plotLayer = NULL,  
  out = "plot",  
  CNscale = "absolute"  
)
```

## Arguments

<code>x</code>	Object of class data frame with rows representing copy-number segment calls. The data frame must contain columns with the following names "chromosome", "start", "end", "segmean", "sample".
<code>y</code>	Object of class data frame with rows representing chromosome boundaries for a genome assembly. The data frame must contain columns with the following names "chromosome", "start", "end" (optional: see details).
<code>genome</code>	Character string specifying a valid UCSC genome (see details).
<code>plot_title</code>	Character string specifying title to display on the plot.
<code>CN_Loss_colour</code>	Character string specifying the colour value of copy number losses.
<code>CN_Gain_colour</code>	Character string specifying the colour value of copy number gains.
<code>x_title_size</code>	Integer specifying the size of the x-axis title.
<code>y_title_size</code>	Integer specifying the size of the y-axis title.
<code>facet_lab_size</code>	Integer specifying the size of the faceted labels plotted.
<code>plotLayer</code>	Valid ggplot2 layer to be added to the plot.
<code>out</code>	Character vector specifying the the object to output, one of "data", "grob", or "plot", defaults to "plot" (see returns).
<code>CNscale</code>	Character string specifying if copy number calls supplied are relative (i.e. copy neutral == 0) or absolute (i.e. copy neutral ==2). One of "relative" or "absolute"

## Details

cnSpec requires the location of chromosome boundaries for a given genome assembly in order to ensure the entire chromosome space is plotted. As a convenience this information is available to cnSpec for the following genomes "hg19", "hg38", "mm9", "mm10", "rn5" and can be retrieved by supplying one of the afore mentioned assemblies via the 'genome' parameter. If a genome assembly is supplied to the 'genome' parameter and is unrecognized cnSpec will attempt to query the UCSC MySQL database for the required information. If chromosome boundary locations are unavailable for a given assembly or if it is desirable to plot a specific region encapsulating the copy number data these boundaries can be supplied to the 'y' parameter which has priority of the parameter 'genome'.

The 'plotLayer' parameter can be used to add an additional layer to the ggplot2 graphic (see vignette).

## Value

One of the following, a list of dataframes containing data to be plotted, a grob object, or a plot.

## Examples

```
cnSpec(LucCNseg, genome="hg19")
```

---

cnView

*Construct copy-number single sample plot*

---

## Description

Given a data frame construct a plot to display raw copy number calls for a single sample.

## Usage

```
cnView(
  x,
  y = NULL,
  z = NULL,
  genome = "hg19",
  chr = "chr1",
  CNscale = "absolute",
  ideogram_txtAngle = 45,
  ideogram_txtSize = 5,
  plotLayer = NULL,
  ideogramLayer = NULL,
  out = "plot",
  segmentColor = NULL
)
```

## Arguments

**x** Object of class data frame with rows representing copy number calls from a single sample. The data frame must contain columns with the following names "chromosome", "coordinate", "cn", and optionally "p\_value" (see details).



y	Object of class data frame with rows representing cytogenetic bands for a chromosome. The data frame must contain columns with the following names "chrom", "chromStart", "chromEnd", "name", "gieStain" for plotting the ideogram (optional: see details).
z	Object of class data frame with row representing copy number segment calls. The data frame must contain columns with the following names "chromosome", "start", "end", "segmean" (optional: see details)
genome	Character string specifying a valid UCSC genome (see details).
chr	Character string specifying which chromosome to plot one of "chr.." or "all"
CNscale	Character string specifying if copy number calls supplied are relative (i.e. copy neutral == 0) or absolute (i.e. copy neutral ==2). One of "relative" or "absolute"
ideogram_txtAngle	Integer specifying the angle of cytogenetic labels on the ideogram subplot.
ideogram_txtSize	Integer specifying the size of cytogenetic labels on the ideogram subplot.
plotLayer	Valid ggplot2 layer to be added to the copy number plot.
ideogramLayer	Valid ggplot2 layer to be added to the ideogram sub-plot.
out	Character vector specifying the the object to output, one of "data", "grob", or "plot", defaults to "plot" (see returns).
segmentColor	Character string specifying the color of segment lines. Used only if Z is not null.

## Details

cnView is able to plot in two modes specified via the 'chr' parameter, these modes are single chromosome view in which an ideogram is displayed and genome view where chromosomes are faceted. For the single chromosome view cytogenetic band information is required giving the coordinate, stain, and name of each band. As a convenience cnView stores this information for the following genomes "hg19", "hg38", "mm9", "mm10", and "rn5". If the genome assembly supplied to the 'genome' parameter is not one of the 5 afore mentioned genome assemblies cnView will attempt to query the UCSC MySQL database to retrieve this information. Alternatively the user can manually supply this information as a data frame to the 'y' parameter, input to the 'y' parameter take precedence of input to 'genome'.

cnView is also able to represent p-values for copy-number calls if they are supplied via the "p\_value" column in the argument supplied to x. The presence of this column in x will set a transparency value to copy-number calls with calls of less significance becoming more transparent.

If it is available cnView can plot copy-number segment calls on top of raw calls supplied to parameter 'x' via the parameter 'z'.

## Value

One of the following, a list of dataframes containing data to be plotted, a grob object, or a plot.

## Examples

```
# Create data
chromosome <- 'chr14'
coordinate <- sort(sample(0:106455000, size=2000, replace=FALSE))
cn <- c(rnorm(300, mean=3, sd=.2), rnorm(700, mean=2, sd=.2), rnorm(1000, mean=3, sd=.2))
data <- as.data.frame(cbind(chromosome, coordinate, cn))

# Plot raw copy number calls
cnView(data, chr='chr14', genome='hg19', ideogram_txtSize=4)
```

compIdent

*Construct identity snp comparison plot***Description**

Given the bam file path, count the number of reads at the 24 SNP locations

**Usage**

```
compIdent(
  x,
  genome,
  target = NULL,
  debug = FALSE,
  mainLayer = NULL,
  covLayer = NULL,
  out = "plot"
)
```

**Arguments**

x	data frame with rows representing samples and column names "sample_name", "bamfile". Columns should correspond to a sample name and a bam file path.
genome	Object of class BSgenome specifying the genome.
target	Object of class data frame containing target locations in 1-base format and containing columns names "chr", "start", "end", "var", "name". Columns should correspond to chromosome, start, end, variant allele, name of location.
debug	Boolean specifying if test datasets should be used for debugging.
mainLayer	Valid ggplot2 layer for altering the main plot.
covLayer	Valid ggplot2 layer for altering the coverage plot.
out	Character vector specifying the the object to output, one of "data", "grob", or "plot", defaults to "plot" (see returns).

**Details**

compIdent is a function designed to compare samples via variant allele frequencies (VAF) at specific sites. By default these sites correspond to 24 identity snps originating from the hg19 assembly however the user can specify alternate sites via the target parameter. To view the 24 identity snp locations use GenVisR::SNPloci.

Samples from the same origin are expected to have similar VAF values however results can skew based on copy number alterations (CNA). The user is expected to ensure no CNA occur at the 24 identity snp sites.

For display and debugging purposes a debug parameter is available which will use predefined data instead of reading in bam files. Note that data in the debug parameter is only available at the aforementioned 24 sites.

**Value**

One of the following, a list of dataframes containing data to be plotted, a grob object, or a plot.

**Examples**

```
# Read in BSgenome object (hg19)
library(BSgenome.Hsapiens.UCSC.hg19)
hg19 <- BSgenome.Hsapiens.UCSC.hg19

# Generate plot
compIdent(genome=hg19, debug=TRUE)
```

covBars

*Construct an overall coverage cohort plot***Description**

Given a matrix construct a plot to display sequencing depth achieved as percentage bars for a cohort of samples.

**Usage**

```
covBars(
  x,
  colour = NULL,
  plot_title = NULL,
  x_title_size = 12,
  y_title_size = 12,
  facet_lab_size = 10,
  plotLayer = NULL,
  out = "plot"
)
```

**Arguments**

<code>x</code>	Object of class matrix with rows representing the sequencing depth (i.e. number of reads) and columns corresponding to each sample in the cohort and elements of the matrix
<code>colour</code>	Character vector specifying colours to represent sequencing depth.
<code>plot_title</code>	Character string specifying the title to display on the plot.
<code>x_title_size</code>	Integer specifying the size of the x-axis title.
<code>y_title_size</code>	Integer specifying the size of the y-axis title.
<code>facet_lab_size</code>	Integer specifying the size of the faceted labels plotted.
<code>plotLayer</code>	Valid ggplot2 layer to be added to the plot.
<code>out</code>	Character vector specifying the the object to output, one of "data", "grob", or "plot", defaults to "plot" (see returns).

**Value**

One of the following, a list of dataframes containing data to be plotted, a grob object, or a plot.

**Examples**

```
# Create data
x <- matrix(sample(100000,500), nrow=50, ncol=10, dimnames=list(0:49,paste0("Sample",1:10)))

# Call plot function
covBars(x)
```

---

cytoGeno	<i>Cytogenetic banding dataset</i>
----------	------------------------------------

---

**Description**

A data set containing cytogenetic band information for all chromosomes in the following genomes "hg38", "hg19", "mm10", "mm9", "rn5", obtained from the UCSC sql database at genome-mysql.cse.ucsc.edu.

**Usage**

```
data(cytoGeno)
```

**Format**

a data frame with 3207 observations and 6 variables

**Value**

Object of class data frame

---

drawPlot	<i>Method drawPlot</i>
----------	------------------------

---

**Description**

Method drawPlot

**Usage**

```
drawPlot(object, ...)

## S4 method for signature 'Clinical'
drawPlot(object, ...)

## S4 method for signature 'Lollipop'
drawPlot(object, ...)

## S4 method for signature 'MutSpectra'
drawPlot(object, ...)

## S4 method for signature 'Rainfall'
drawPlot(object, ...)

## S4 method for signature 'Waterfall'
drawPlot(object, ...)
```

**Arguments**

object            Object of class Waterfall, MutSpectra, or Clinical  
 ...                additional arguments to passed

**Details**

The drawPlot method is used to draw plots created by GenVisR plot constructor functions.

---

genCov                            *Construct a region of interest coverage plot*

---

**Description**

Given a list of data frames construct a sequencing coverage view over a region of interest.

**Usage**

```
genCov(
  x,
  txdb,
  gr,
  genome,
  reduce = FALSE,
  gene_colour = NULL,
  gene_name = "Gene",
  gene_plotLayer = NULL,
  label_bgFill = "black",
  label_txtFill = "white",
  label_borderFill = "black",
  label_txtSize = 10,
  lab2plot_ratio = c(1, 10),
  cov_colour = "blue",
  cov_plotType = "point",
  cov_plotLayer = NULL,
  base = c(10, 2, 2),
  transform = c("Intron", "CDS", "UTR"),
  gene_labelTranscript = TRUE,
  gene_labelTranscriptSize = 4,
  gene_isoformSel = NULL,
  out = "plot",
  subsample = FALSE
)
```

**Arguments**

x                                Named list with list elements containing data frames representing samples. Data frame rows should represent read pileups observed in sequencing data. Data frame column names must include "end" and "cov" corresponding to the base end position and coverage of a pileup respectively. Data within data frames must be on the same chromosome as the region of interest, see details!

txdb	Object of class TxDb giving transcription meta data for a genome assembly. See Bioconductor annotation packages.
gr	Object of class GRanges specifying the region of interest and corresponding to a single gene. See Bioconductor package GRanges.
genome	Object of class BSgenome specifying the genome sequence of interest. See Bioconductor annotation packages.
reduce	Boolean specifying whether to collapse gene isoforms within the region of interest into one representative transcript. Experimental use with caution!
gene_colour	Character string specifying the colour of the gene to be plotted in the gene track.
gene_name	Character string specifying the name of the gene or region of interest.
gene_plotLayer	Valid ggplot2 layer to be added to the gene sub-plot.
label_bgFill	Character string specifying the desired background colour of the track labels.
label_txtFill	Character string specifying the desired text colour of the track labels.
label_borderFill	Character string specifying the desired border colour of the track labels.
label_txtSize	Integer specifying the size of the text within the track labels.
lab2plot_ratio	Numeric vector of length 2 specifying the ratio of track labels to plot space.
cov_colour	Character string specifying the colour of the data in the coverage plots.
cov_plotType	Character string specifying one of "line", "bar" or "point". Changes the ggplot2 geom which constructs the data display.
cov_plotLayer	Valid ggplot2 layer to be added to the coverage sub-plots.
base	Numeric vector of log bases to transform the data corresponding to the elements supplied to the variable transform See details.
transform	Character vector specifying what objects to log transform, accepts "Intron", "CDS", and "UTR" See details.
gene_labelTranscript	Boolean specifying whether to plot the transcript names in the gene plot.
gene_labelTranscriptSize	Integer specifying the size of the transcript name text in the gene plot.
gene_isoformSel	Character vector specifying the names (from the txdb object) of isoforms within the region of interest to display.
out	Character vector specifying the object to output, one of "data", "grob", or "plot", defaults to "plot" (see returns).
subsample	Boolean value specifying whether to reduce the provided coverage data to a subset of approximately 1000 points. Used to generate sparse plots that use less disk space and are faster to render.

## Details

genCov is a function designed construct a series of tracks based on a TxDb object giving transcript features, and coverage data supplied to parameter 'x'. The function will look at a region of interest specified by the argument supplied to gr and plot transcript features and the corresponding coverage information. The argument supplied to 'genome' enables gc content within genomic features to be calculated and displayed. The argument supplied to x must contain data on the same chromosome as the region of interest specified in the parameter 'gr'!

Typically, introns of a transcript are much larger than exons, while exons are sometimes of greater interest. To address this, `genCov` will by default scale the x-axis to expand track information according to region type: coding sequence (CDS), untranslated region (UTR), or intron / intergenic (Intron). The amount by which each region is scaled is controlled by the `'base'` and `'transform'` arguments. `'transform'` specifies which regions to scale, and `'base'` corresponds to the log base transform to apply to those regions. To keep one or more region types from being scaled, omit the corresponding entries from the `'base'` and `'transform'` vectors.

## Value

One of the following, a list of dataframes containing data to be plotted, a grob object, or a plot.

## Examples

```
# Load transcript meta data
library(TxDb.Hsapiens.UCSC.hg19.knownGene)
txdb <- TxDb.Hsapiens.UCSC.hg19.knownGene

# Load BSgenome
library(BSgenome.Hsapiens.UCSC.hg19)
genome <- BSgenome.Hsapiens.UCSC.hg19

# Define a region of interest
gr <- GRanges(seqnames=c("chr10"),
              ranges=IRanges(start=c(89622195), end=c(89729532)), strand=strand(c("+")))

# Create Data for input
start <- c(89622194:89729524)
end <- c(89622195:89729525)
chr <- 10
cov <- c(rnorm(100000, mean=40), rnorm(7331, mean=10))
cov_input_A <- as.data.frame(cbind(chr, start, end, cov))

start <- c(89622194:89729524)
end <- c(89622195:89729525)
chr <- 10
cov <- c(rnorm(50000, mean=40), rnorm(7331, mean=10), rnorm(50000, mean=40))
cov_input_A <- as.data.frame(cbind(chr, start, end, cov))

# Define the data as a list
data <- list("Sample A"=cov_input_A)

# Call genCov
genCov(data, txdb, gr, genome, gene_labelTranscriptSize=3)
```

---

geneViz

*Construct a gene-features plot*

---

## Description

Given a `GRanges` object specifying a region of interest, plot genomic features within that region.

**Usage**

```
geneViz(
  txdb,
  gr,
  genome,
  reduce = FALSE,
  gene_colour = NULL,
  base = c(10, 2, 2),
  transform = c("Intron", "CDS", "UTR"),
  isoformSel = NULL,
  labelTranscript = TRUE,
  labelTranscriptSize = 4,
  plotLayer = NULL
)
```

**Arguments**

txdb	Object of class TxDb giving transcription meta data for a genome assembly. See Bioconductor annotation packages.
gr	Object of class GRanges specifying the region of interest and corresponding to a single gene. See Bioconductor package GRanges.
genome	Object of class BSgenome specifying the genome sequence of interest. See Bioconductor annotation packages.
reduce	Boolean specifying whether to collapse gene isoforms within the region of interest into one representative transcript. Experimental use with caution!
gene_colour	Character string specifying the colour of the gene to be plotted.
base	Numeric vector of log bases to transform the data corresponding to the elements supplied to the variable transform See details.
transform	Character vector specifying what objects to log transform, accepts "Intron", "CDS", and "UTR" See details.
isoformSel	Character vector specifying the names (from the txdb object) of isoforms within the region of interest to display.
labelTranscript	Boolean specifying whether to plot the transcript names in the gene plot.
labelTranscriptSize	Integer specifying the size of the transcript name text in the gene plot.
plotLayer	Valid ggplot2 layer to be added to the gene plot.

**Details**

geneViz is an internal function which will output a list of three elements. As a convenience the function is exported however to obtain the plot from geneViz the user must call the first element of the list. geneViz is intended to plot gene features within a single gene with boundaries specified by the GRanges object, plotting more than one gene is advised against.

Typically, introns of a transcript are much larger than exons, while exons are sometimes of greater interest. To address this, genCov will by default scale the x-axis to expand track information according to region type: coding sequence (CDS), untranslated region (UTR), or intron / intergenic (Intron). The amount by which each region is scaled is controlled by the 'base' and 'transform' arguments. 'transform' specifies which regions to scale, and 'base' corresponds to the log base transform to apply to those regions. To keep one or more region types from being scaled, omit the corresponding entries from the 'base' and 'transform' vectors.



**Value**

object of class list with list elements containing a ggplot object, the gene features within the plot as a data frame, and mapping information of the gene features within the ggplot object.

**Examples**

```
# need transcript data for reference
library(TxDb.Hsapiens.UCSC.hg19.knownGene)
txdb <- TxDb.Hsapiens.UCSC.hg19.knownGene

# need a biostrings object for reference
library(BSgenome.Hsapiens.UCSC.hg19)
genome <- BSgenome.Hsapiens.UCSC.hg19

# need Granges object
gr <- GRanges(seqnames=c("chr10"),
              ranges=IRanges(start=c(89622195), end=c(89729532)), strand=strand(c("+")))

# Plot the graphic
geneViz(txdb, gr, genome)
```

---

 GenVisR

*GenVisR*


---

**Description**

A visualization library designed to make publications quality figures for genomic datasets.

**References**

[GenVisR: Genomic Visualizations in R](#)

**See Also**

[GenVisR github page](#)

[GenVisR bioconductor page](#)

---

 getData

*Method getData*


---

**Description**

Method getData

Helper function to get data from classes

Helper function to getData from classes, under development!!!

Helper function to get data from classes

Helper function to getData from classes, under development!!!

**Usage**

```

getData(object, ...)

## S4 method for signature 'Clinical'
getData(object, ...)

## S4 method for signature 'ClinicalData'
getData(object, ...)

.getData_Lollipop(object, name = NULL, index = NULL, ...)

## S4 method for signature 'LollipopData'
getData(object, name = NULL, index = NULL, ...)

## S4 method for signature 'Lollipop'
getData(object, name = NULL, index = NULL, ...)

.getData_MutSpectra(object, name = NULL, index = NULL, ...)

## S4 method for signature 'MutSpectraPrimaryData'
getData(object, name = NULL, index = NULL, ...)

## S4 method for signature 'MutSpectra'
getData(object, name = NULL, index = NULL, ...)

.getData_Rainfall(object, name = NULL, index = NULL, ...)

## S4 method for signature 'RainfallPrimaryData'
getData(object, name = NULL, index = NULL, ...)

## S4 method for signature 'Rainfall'
getData(object, name = NULL, index = NULL, ...)

.getData_waterfall(object, name = NULL, index = NULL, ...)

## S4 method for signature 'WaterfallData'
getData(object, name = NULL, index = NULL, ...)

## S4 method for signature 'Waterfall'
getData(object, name = NULL, index = NULL, ...)

```

**Arguments**

object	Object of class Clinical,
...	additional arguments to passed
name	String corresponding to the slot for which to extract data from.
index	Integer specifying the slot for which to extract data from.

**Details**

The `getData` method is an accessor function used to access data held in GenVisR objects.

---

getDescription	<i>Method getDescription</i>
----------------	------------------------------

---

**Description**

Method getDescription

**Usage**

```
getDescription(object, ...)

## S4 method for signature 'VEP_Virtual'
getDescription(object, ...)

## S4 method for signature 'VEP'
getDescription(object, ...)
```

**Arguments**

object	Object of class VEP
...	additional arguments to passed

---

getGrob	<i>Method getGrob</i>
---------	-----------------------

---

**Description**

Method getGrob

**Usage**

```
getGrob(object, ...)

## S4 method for signature 'LollipopPlots'
getGrob(object, index = 1, ...)

## S4 method for signature 'Lollipop'
getGrob(object, index = 1, ...)

## S4 method for signature 'MutSpectraPlots'
getGrob(object, index = 1, ...)

## S4 method for signature 'MutSpectra'
getGrob(object, index = 1, ...)

## S4 method for signature 'RainfallPlots'
getGrob(object, index = 1, ...)

## S4 method for signature 'Rainfall'
```

```

getGrob(object, index = 1, ...)

## S4 method for signature 'WaterfallPlots'
getGrob(object, index = 1, ...)

## S4 method for signature 'Waterfall'
getGrob(object, index = 1, ...)

```

### Arguments

object	Object of clas MutSpectra
...	additional arguments to passed
index	integer specifying the plot index to extract

---

getHeader	<i>Method getHeader</i>
-----------	-------------------------

---

### Description

Method getHeader

### Usage

```

getHeader(object, ...)

## S4 method for signature 'VEP_Virtual'
getHeader(object, ...)

## S4 method for signature 'VEP'
getHeader(object, ...)

```

### Arguments

object	Object of class VEP
...	additional arguments to passed

---

getMeta	<i>Method getMeta</i>
---------	-----------------------

---

### Description

Method getMeta

**Usage**

```

getMeta(object, ...)

## S4 method for signature 'GMS_Virtual'
getMeta(object, ...)

## S4 method for signature 'GMS'
getMeta(object, ...)

## S4 method for signature 'MutationAnnotationFormat_Virtual'
getMeta(object, ...)

## S4 method for signature 'MutationAnnotationFormat'
getMeta(object, ...)

## S4 method for signature 'VEP_Virtual'
getMeta(object, ...)

## S4 method for signature 'VEP'
getMeta(object, ...)

```

**Arguments**

object	Object of class VEP, GMS, or MutationAnnotationFormat
...	additional arguments to passed

---

getMutation	<i>Method getMutation</i>
-------------	---------------------------

---

**Description**

Method getMutation

**Usage**

```

getMutation(object, ...)

## S4 method for signature 'GMS_Virtual'
getMutation(object, ...)

## S4 method for signature 'GMS'
getMutation(object, ...)

## S4 method for signature 'MutationAnnotationFormat_Virtual'
getMutation(object, ...)

## S4 method for signature 'MutationAnnotationFormat'
getMutation(object, ...)

## S4 method for signature 'VEP_Virtual'

```

```

getMutation(object, ...)

## S4 method for signature 'VEP'
getMutation(object, ...)

```

**Arguments**

object	Object of class VEP, GMS, or MutationAnnotationFormat
...	additional arguments to passed

---

getPath	<i>Method getPath</i>
---------	-----------------------

---

**Description**

Method getPath

**Usage**

```

getPath(object, ...)

## S4 method for signature 'GMS'
getPath(object, ...)

## S4 method for signature 'MutationAnnotationFormat'
getPath(object, ...)

## S4 method for signature 'VEP'
getPath(object, ...)

```

**Arguments**

object	Object of class VEP, GMS, or MutationAnnotationFormat
...	additional arguments to passed

---

getPosition	<i>Method getPosition</i>
-------------	---------------------------

---

**Description**

Method getPosition

**Usage**

```

getPosition(object, ...)

## S4 method for signature 'GMS_Virtual'
getPosition(object, ...)

## S4 method for signature 'GMS'
getPosition(object, ...)

## S4 method for signature 'MutationAnnotationFormat_Virtual'
getPosition(object, ...)

## S4 method for signature 'MutationAnnotationFormat'
getPosition(object, ...)

## S4 method for signature 'VEP_Virtual'
getPosition(object, ...)

## S4 method for signature 'VEP'
getPosition(object, ...)

```

**Arguments**

object	Object of class VEP, GMS, or MutationAnnotationFormat
...	additional arguments to passed

---

getSample	<i>Method getSample</i>
-----------	-------------------------

---

**Description**

Method getSample

**Usage**

```

getSample(object, ...)

## S4 method for signature 'GMS_Virtual'
getSample(object, ...)

## S4 method for signature 'GMS'
getSample(object, ...)

## S4 method for signature 'MutationAnnotationFormat_Virtual'
getSample(object, ...)

## S4 method for signature 'MutationAnnotationFormat'
getSample(object, ...)

## S4 method for signature 'VEP_Virtual'

```

```
getSample(object, ...)

## S4 method for signature 'VEP'
getSample(object, ...)
```

**Arguments**

object	Object of class VEP, GMS, or MutationAnnotationFormat
...	additional arguments to passed

---

getVersion	<i>Method getVersion</i>
------------	--------------------------

---

**Description**

Method getVersion

**Usage**

```
getVersion(object, ...)

## S4 method for signature 'GMS'
getVersion(object, ...)

## S4 method for signature 'MutationAnnotationFormat'
getVersion(object, ...)

## S4 method for signature 'VEP'
getVersion(object, ...)
```

**Arguments**

object	Object of class VEP, GMS, or MutationAnnotationFormat
...	additional arguments to passed

---

GMS-class	<i>Class GMS</i>
-----------	------------------

---

**Description**

An S4 class for Genome Modeling System annotation files, under development!!!

**Usage**

```
GMS(path, data = NULL, version = 4, verbose = FALSE)
```



**Arguments**

path	String specifying the path to a GMS annotation file. Can accept wildcards if multiple GMS annotation files exist (see details).
data	data.table object storing a GMS annotation file. Overrides "path" if specified.
version	String specifying the version of the GMS files, Defaults to version 4.
verbose	Boolean specifying if progress should be reported while reading in the GMS files.

**Details**

When specifying a path to a GMS annotation file the option exist to either specify the full path to an annotation file or to use wildcards to specify multiple files. When specifying a full path the initializer will check if a column named "sample" containing the relevant sample for each row exists. If such a column is not found the initializer will assume this file corresponds to only one sample and populate a sample column accordingly. Alternatively if multiple files are specified at once using a wildcard, the initializer will aggregate all the files and use the file names minus any extension to populate sample names. The version defaults to 4 which is the default value of the GMS annotator. This value will need to be changed only if files were created using a different GMS annotator version.

**Slots**

path	Character string specifying the paths of the GMS files read in.
version	Numeric value specifying the version of the GMS annotation files.
gmsObject	gms object which inherits from gms_Virtual class.

**See Also**

[Waterfall](#)  
[MutSpectra](#)

---

GMS\_v4-class

*Class GMS\_v4*


---

**Description**

An S4 class to represent data in gms annotation version 4, inherits from the GMS\_Virtual class.

**Usage**

```
GMS_v4(gmsData)
```

**Arguments**

gmsData	data.table object containing a gms annotation file conforming to the version 4 specifications.
---------	--

**Slots**

position data.table object containing column names "chromosome\_name", "start", "stop".

mutation data.table object containing column names "reference", "variant", "trv\_type".

sample data.table object containing columns names "sample".

meta data.table object containing meta data.

---

GMS_Virtual-class	<i>Class GMS_Virtual</i>
-------------------	--------------------------

---

**Description**

An S4 class to act as a virtual class for GMS version sub-classes.

**Slots**

position data.table object holding genomic positions.

mutation data.table object holding mutation status data.

sample data.table object holding sample data.

meta data.table object holding all other meta data.

---

HCC1395_Germline	<i>Germline Calls</i>
------------------	-----------------------

---

**Description**

A data set containing downsampled Germline calls originating from the HCC1395 breast cancer cell line.

**Usage**

```
data(HCC1395_Germline)
```

**Format**

a data frame with 9200 observations and 5 variables

**Value**

Object of class data frame

---

HCC1395\_N

*Normal BAM*

---

**Description**

A data set containing read pileups intersecting 24 identity snp locations from GenVisR::SNPloci. Pileups are from downsampled bams and originate from normal tissue corresponding to the HCC1395 breast cancer cell line.

**Usage**

```
data(HCC1395_N)
```

**Format**

a data frame with 59 observations and 6 variables

**Value**

Object of class list

---

HCC1395\_T

*Tumor BAM*

---

**Description**

A data set containing read pileups intersecting 24 identity snp locations from GenVisR::SNPloci. Pileups are from downsampled bams and originate from tumor tissue corresponding to the HCC1395 breast cancer cell line.

**Usage**

```
data(HCC1395_T)
```

**Format**

a data frame with 52 observations and 6 variables

**Value**

Object of class list

---

hg19chr	<i>hg19 chromosome boundaries</i>
---------	-----------------------------------

---

**Description**

A data set containing chromosome boundaries corresponding to hg19.

**Usage**

```
data(hg19chr)
```

**Format**

a data frame with 24 observations and 3 variables

**Value**

Object of class data frame

---

ideoView	<i>Construct an ideogram</i>
----------	------------------------------

---

**Description**

Given a data frame with cytogenetic information, construct an ideogram.

**Usage**

```
ideoView(
  x,
  chromosome = "chr1",
  txtAngle = 45,
  txtSize = 5,
  plotLayer = NULL,
  out = "plot"
)
```

**Arguments**

x	Object of class data frame with rows representing cytogenetic bands. The data frame must contain the following column names "chrom", "chromStart", "chromEnd", "name", "gieStain"
chromosome	Character string specifying which chromosome from the "chrom" column in the argument supplied to parameter x to plot.
txtAngle	Integer specifying the angle of text labeling cytogenetic bands.
txtSize	Integer specifying the size of text labeling cytogenetic bands.
plotLayer	additional ggplot2 layers for the ideogram
out	Character vector specifying the the object to output, one of "data", "grob", or "plot", defaults to "plot" (see returns).

**Details**

ideoView is a function designed to plot cytogenetic band information. Modifications to the graphic object can be made via the 'plotLayer' parameter, see vignette for details.

**Value**

One of the following, a list of dataframes containing data to be plotted, a grob object, or a plot.

**Examples**

```
# Obtain cytogenetic information for the genome of interest from attached
# data set cytoGeno
data <- cytoGeno[cytoGeno$genome == 'hg38',]

# Call ideoView for chromosome 1
ideoView(data, chromosome='chr1', txtSize=4)
```

lohSpec

*Plot LOH data***Description**

Construct a graphic visualizing Loss of Heterozygosity in a cohort

**Usage**

```
lohSpec(
  x = NULL,
  path = NULL,
  fileExt = NULL,
  y = NULL,
  genome = "hg19",
  gender = NULL,
  step = 1e+06,
  window_size = 2500000,
  normal = 0.5,
  colourScheme = "inferno",
  plotLayer = NULL,
  method = "slide",
  out = "plot"
)
```

**Arguments**

- |      |   |
|------|---|
| x    | object of class data frame with rows representing germline calls. The data frame must contain columns with the following names "chromosome", "position", "n_vaf", "t_vaf", "sample". required if path is set to NULL (see details). vaf should range from 0-1.  |
| path | Character string specifying the path to a directory containing germline calls for each sample. Germline calls are expected to be stored as tab-separated files which contain the following column names "chromosome", "position", "n_vaf", "t_vaf", and "sample". required if x is set to null (see details). |

fileExt	Character string specifying the file extensions of files within the path specified. Required if argument is supplied to path (see details).
y	Object of class data frame with rows representing chromosome boundaries for a genome assembly. The data frame must contain columns with the following names "chromosome", "start", "end" (optional: see details).
genome	Character string specifying a valid UCSC genome (see details).
gender	Character vector of length equal to the number of samples, consisting of elements from the set "M", "F". Used to suppress the plotting of allosomes where appropriate.
step	Integer value specifying the step size (i.e. the number of base pairs to move the window). required when method is set to slide (see details).
window_size	Integer value specifying the size of the window in base pairs in which to calculate the mean Loss of Heterozygosity (see details).
normal	Numeric value within the range 0-1 specifying the expected normal variant allele frequency to be used in Loss of Heterozygosity calculations. defaults to .50%
colourScheme	Character vector specifying the colour scale to use from the viridis package. One of "viridis", "magma", "plasma", or "inferno".
plotLayer	Valid ggplot2 layer to be added to the plot.
method	character string specifying the approach to be used for displaying Loss of Heterozygosity, one of "tile" or "slide" (see details).
out	Character vector specifying the the object to output, one of "data", "grob", or "plot", defaults to "plot" (see returns).

## Details

lohSpec is intended to plot the loss of heterozygosity (LOH) within a sample. As such lohSpec expects input data to contain only LOH calls. Input can be supplied as a single data frame given to the argument x with rows containing germline calls and variables giving the chromosome, position, normal variant allele frequency, tumor variant allele frequency, and the sample. In lieu of this format a series of .tsv files can be supplied via the path and fileExt arguments. If this method is choosen samples will be infered from the file names. In both cases columns containing the variant allele frequency for normal and tumor samples should range from 0-1. Two methods exist to calculate and display LOH events. If the method is set to "tile" mean LOH is calculated based on the window\_size argument with windows being placed next to each other. If the method is set to slide the widnow will slide and calculate the LOH based on the step parameter. In order to ensure the entire chromosome is plotted lohSpec requiries the location of chromosome boundaries for a given genome assembly. As a convenience this information is available for the following genomes "hg19", "hg38", "mm9", "mm10", "rn5" and can be tetrieved by supplying one of the afore mentioned assemblies via the 'genome' paramter. If an argument is supplied to the 'genome' parameter and is unrecognized a query to the UCSC MySQL database will be attempted to obtain the required information. If chromosome boundary locations are unavailable for a given assembly this information can be supplied to the 'y' parameter which has priority over the 'genome' parameter.

## Value

One of the following, a list of dataframes containing data to be plotted, a grob object, or a plot.

## Examples

```
# plot loh within the example dataset
lohSpec(x=HCC1395_Germline)
```

lohView

*Construct LOH chromosome plot***Description**

Given a data frame construct a plot to display Loss of Heterozygosity for specific chromosomes.

**Usage**

```
lohView(
  x,
  y = NULL,
  genome = "hg19",
  chr = "chr1",
  ideogram_txtAngle = 45,
  ideogram_txtSize = 5,
  plotLayer = NULL,
  ideogramLayer = NULL,
  out = "plot"
)
```

**Arguments**

x	object of class data frame with rows representing Heterozygous Germline calls. The data frame must contain columns with the following names "chromosome", "position", "n_vaf", "t_vaf", "sample".
y	Object of class data frame with rows representing cytogenetic bands for a chromosome. The data frame must contain columns with the following names "chrom", "chromStart", "chromEnd", "name", "gieStain" for plotting the ideogram (optional: see details).
genome	Character string specifying a valid UCSC genome (see details).
chr	Character string specifying which chromosome to plot one of "chr..." or "all"
ideogram_txtAngle	Integer specifying the angle of cytogenetic labels on the ideogram subplot.
ideogram_txtSize	Integer specifying the size of cytogenetic labels on the ideogram subplot.
plotLayer	Valid ggplot2 layer to be added to the copy number plot.
ideogramLayer	Valid ggplot2 layer to be added to the ideogram sub-plot.
out	Character vector specifying the the object to output, one of "data", "grob", or "plot", defaults to "plot" (see returns).

**Details**

lohView is able to plot in two modes specified via the 'chr' parameter, these modes are single chromosome view in which an ideogram is displayed and genome view where chromosomes are faceted. For the single chromosome view cytogenetic band information is required giving the coordinate, stain, and name of each band. As a convenience GenVisR stores this information for the following genomes "hg19", "hg38", "mm9", "mm10", and "rn5". If the genome assembly supplied to the 'genome' parameter is not one of the 5 afore mentioned genome assemblies GenVisR will

attempt to query the UCSC MySQL database to retrieve this information. Alternatively the user can manually supply this information as a data frame to the 'y' parameter, input to the 'y' parameter take precedence of input to 'genome'.

A word of caution, users are advised to only use heterozygous germline calls in input to 'x', failure to do so may result in a misleading visual!

### Value

One of the following, a list of dataframes containing data to be plotted, a grob object, or a plot.

### Examples

```
# Plot loh for chromosome 5
lohView(HCC1395_Germline, chr='chr5', genome='hg19', ideogram_txtSize=4)
```

---

lolliplot	<i>Construct a lolliplot</i>
-----------	------------------------------

---

### Description

This function has been removed, please use Lolliplot() (capital L) instead!

### Usage

```
lolliplot()
```

---

Lolliplot-class	<i>Class Lolliplot</i>
-----------------	------------------------

---

### Description

An S4 class for the lolliplot object, under development!!!

### Usage

```
Lolliplot(
  input,
  transcript = NULL,
  species = "hsapiens",
  host = "www.ensembl.org",
  txdb = NULL,
  BSgenome = NULL,
  emphasize = NULL,
  DomainPalette = NULL,
  MutationPalette = NULL,
  labelAA = TRUE,
  plotALayers = NULL,
  plotBLayers = NULL,
  sectionHeights = NULL,
  verbose = FALSE
)
```





```

# mode 2, amino acid changes are present

keep <- c("Chromosome", "Start_Position", "End_Position", "Reference_Allele",
          "Tumor_Seq_Allele2", "Tumor_Sample_Barcode", "Gene", "Variant_Classification",
          "Transcript_ID", "HGVS")
dataset.mode2 <- dataset[,keep]
colnames(dataset.mode2) <- c("chromosome", "start", "stop", "reference", "variant",
                             "sample", "gene", "consequence", "transcript", "proteinCoord")

# run Lollipop

object <- Lollipop(dataset.mode1, transcript="ENST00000263967",
                   species="hsapiens", txd=txdb, BSgenome=BSgenome)
object <- Lollipop(dataset.mode2, transcript="ENST00000263967",
                   species="hsapiens")

```

---

LucCNseg

*Truncated CN segments*

---

### Description

A data set in long format containing Copy Number segments for 4 samples corresponding to "lung cancer" from Govindan et al. Cell. 2012, PMID:22980976

### Usage

```
data(LucCNseg)
```

### Format

a data frame with 3336 observations and 6 variables

### Value

Object of class data frame

---

MutationAnnotationFormat-class

*Class MutationAnnotationFormat*

---

### Description

An S4 class acting as a container for MutationAnnotationFormat version sub-classes, under development!!!

### Usage

```
MutationAnnotationFormat(path, version = "auto", verbose = FALSE)
```

**Arguments**

path	String specifying the path to a MAF file.
version	String specifying the version of the MAF file, if set to auto the version will be obtained from the header in the MAF file.
verbose	Boolean specifying if progress should be reported while reading in the MAF file.

**Slots**

path	Character string specifying the path of the MAF file read in.
version	Numeric value specifying the version of the MAF file.
mafObject	MutationAnnotationFormat object which inherits from MutationAnnotationFormat_Virtual class.

**See Also**

[Waterfall](#)  
[MutSpectra](#)

---

MutationAnnotationFormat\_v1.0-class

*Class MutationAnnotationFormat\_v1.0*

---

**Description**

An S4 class to represent data in mutation annotation format version 1.0, inherits from the MutationAnnotationFormat\_Virtual class.

**Usage**

```
MutationAnnotationFormat_v1.0(mafData)
```

**Arguments**

mafData	data.table object containing a maf file conforming to the version 1.0 specification.
---------	--

**Slots**

position	data.table object containing column names "Chromosome", "Start_Position", "End_Position", "Strand".
mutation	data.table object containing column names "Variant_Classification", "Variant_Type", "Reference_Allele", "Tumor_Seq_Allele1", "Tumor_Seq_Allele2".
sample	data.table object containing columns names "Tumor_Sample_Barcode".
meta	data.table object containing meta data.

MutationAnnotationFormat\_v2.0-class

*Class MutationAnnotationFormat\_v2.0*

---

### Description

An S4 class to represent data in mutation annotation format version 2.0, inherits from the MutationAnnotationFormat\_Virtual class.

### Usage

```
MutationAnnotationFormat_v2.0(mafData)
```

### Arguments

mafData            data.table object containing a maf file conforming to the version 2.0 specification.

### Slots

position data.table object containing column names "Chromosome", "Start\_Position", "End\_Position", "Strand".

mutation data.table object containing column names "Variant\_Classification", "Variant\_Type", "Reference\_Allele", "Tumor\_Seq\_Allele1", "Tumor\_Seq\_Allele2".

sample data.table object containing columns names "Tumor\_Sample\_Barcode".

meta data.table object containing meta data.

---

MutationAnnotationFormat\_v2.1-class

*Class MutationAnnotationFormat\_v2.1*

---

### Description

An S4 class to represent data in mutation annotation format version 2.1, inherits from the MutationAnnotationFormat\_Virtual class.

### Usage

```
MutationAnnotationFormat_v2.1(mafData)
```

### Arguments

mafData            data.table object containing a maf file conforming to the version 2.1 specification.

**Slots**

`position` data.table object containing column names "Chromosome", "Start\_Position", "End\_Position", "Strand".

`mutation` data.table object containing column names "Variant\_Classification", "Variant\_Type", "Reference\_Allele", "Tumor\_Seq\_Allele1", "Tumor\_Seq\_Allele2".

`sample` data.table object containing columns names "Tumor\_Sample\_Barcode".

`meta` data.table object containing meta data.

---

MutationAnnotationFormat\_v2.2-class

*Class MutationAnnotationFormat\_v2.2*

---

**Description**

An S4 class to represent data in mutation annotation format version 2.2, inherits from the MutationAnnotationFormat\_Virtual class.

**Usage**

`MutationAnnotationFormat_v2.2(mafData)`

**Arguments**

`mafData` data.table object containing a maf file conforming to the version 2.2 specification.

**Slots**

`position` data.table object containing column names "Chromosome", "Start\_Position", "End\_Position", "Strand".

`mutation` data.table object containing column names "Variant\_Classification", "Variant\_Type", "Reference\_Allele", "Tumor\_Seq\_Allele1", "Tumor\_Seq\_Allele2".

`sample` data.table object containing columns names "Tumor\_Sample\_Barcode".

`meta` data.table object containing meta data.

---

MutationAnnotationFormat\_v2.3-class

*Class MutationAnnotationFormat\_v2.3*

---

**Description**

An S4 class to represent data in mutation annotation format version 2.3, inherits from the MutationAnnotationFormat\_Virtual class.

**Usage**

`MutationAnnotationFormat_v2.3(mafData)`

**Arguments**

mafData data.table object containing a maf file conforming to the version 2.3 specification.

**Slots**

position data.table object containing column names "Chromosome", "Start\_Position", "End\_Position", "Strand".

mutation data.table object containing column names "Variant\_Classification", "Variant\_Type", "Reference\_Allele", "Tumor\_Seq\_Allele1", "Tumor\_Seq\_Allele2".

sample data.table object containing columns names "Tumor\_Sample\_Barcode".

meta data.table object containing meta data.

---

MutationAnnotationFormat\_v2.4-class

*Class MutationAnnotationFormat\_v2.4*

---

**Description**

An S4 class to represent data in mutation annotation format version 2.4, inherits from the MutationAnnotationFormat\_Virtual class.

**Usage**

MutationAnnotationFormat\_v2.4(mafData)

**Arguments**

mafData data.table object containing a maf file conforming to the version 2.4 specification.

**Slots**

position data.table object containing column names "Chromosome", "Start\_Position", "End\_Position", "Strand".

mutation data.table object containing column names "Variant\_Classification", "Variant\_Type", "Reference\_Allele", "Tumor\_Seq\_Allele1", "Tumor\_Seq\_Allele2".

sample data.table object containing columns names "Tumor\_Sample\_Barcode".

meta data.table object containing meta data.

---

MutationAnnotationFormat\_Virtual-class  
*Class MutationAnnotationFormat\_Virtual*

---

### Description

An S4 class to act as a virtual class for MutationAnnotationFormat version sub-classes.

### Slots

position data.table object holding genomic positions.  
 mutation data.table object holding mutation status data.  
 sample data.table object holding sample data.  
 meta data.table object holding all other meta data.

---

MutSpectra-class      *Class MutSpectra*

---

### Description

An S4 class for the MutSpectra plot object, under development!!!

### Usage

```
MutSpectra(
  object,
  BSgenome = NULL,
  sorting = NULL,
  palette = NULL,
  clinical = NULL,
  sectionHeights = NULL,
  sampleNames = TRUE,
  verbose = FALSE,
  plotALayers = NULL,
  plotBLayers = NULL,
  plotCLayers = NULL
)
```

### Arguments

object	Object of class MutationAnnotationFormat, GMS, VEP.
BSgenome	Object of class BSgenome, used to extract reference bases if not supplied by the file format.
sorting	Character vector specifying how samples should be ordered in the plot, one of "mutation", "sample", or a vector of length equal to the number of samples explicitly providing the order of samples.
palette	Character vector specifying the colors used for encoding transitions and transversions, should be of length 6. If NULL a default palette will be used.

<code>clinical</code>	Object of class <code>Clinical</code> , used for adding a clinical data subplot.
<code>sectionHeights</code>	Numeric vector specifying relative heights of each plot section, should sum to one. Expects a value for each section.
<code>sampleNames</code>	Boolean specifying if samples should be labeled on the plot.
<code>verbose</code>	Boolean specifying if status messages should be reported
<code>plotALayers</code>	list of <code>ggplot2</code> layers to be passed to the frequency plot.
<code>plotBLayers</code>	list of <code>ggplot2</code> layers to be passed to the proportion plot.
<code>plotCLayers</code>	list of <code>ggplot2</code> layers to be passed to the clinical plot.

### Slots

<code>PlotA</code>	<code>gtable</code> object for the mutation frequencies.
<code>PlotB</code>	<code>gtable</code> object for the mutation proportions.
<code>PlotC</code>	<code>gtable</code> object for clinical data sub-plot.
<code>Grob</code>	<code>gtable</code> object for the arranged plot.
<code>primaryData</code>	<code>data.table</code> object storing the primary data, should have column names <code>sample</code> , <code>mutation</code> , <code>frequency</code> , <code>proportion</code> .
<code>ClinicalData</code>	<code>data.table</code> object storing the data used to plot the clinical sub-plot.

---

PIK3CA

*Subset MAF file for PIK3CA gene*

---

### Description

A data set originating from the open access TCGA data (6c93f518-1956-4435-9806-37185266d248), the data set is composed of mutations for the PIK3CA gene for breast cancer. This is primarily intended to test the `Lollipop()` function.

### Usage

```
data(PIK3CA)
```

### Format

a data frame with 361 observations and 19 variables

### Value

Object of class `data frame`



---

Rainfall-class	<i>Class Rainfall</i>
----------------	-----------------------

---

### Description

An S4 class for the Rainfall plot object, under development!!!

### Usage

```
Rainfall(
  object,
  BSgenome = NULL,
  palette = NULL,
  sectionHeights = NULL,
  chromosomes = NULL,
  sample = NULL,
  pointSize = NULL,
  verbose = FALSE,
  plotALayers = NULL,
  plotBLayers = NULL
)
```

### Arguments

object	Object of class MutationAnnotationFormat, GMS, VEP.
BSgenome	Object of class BSgenome to extract genome wide chromosome coordinates
palette	Character vector specifying colors used for encoding transitions and transversions , should be of length 7. If NULL a default palette will be used.
sectionHeights	Numeric vector specifying relative heights of each plot section, should sum to one. Expects a value for each section.
chromosomes	Character vector specifying chromosomes for which to plot
sample	Character vector specifying the samples for which to plot.
pointSize	numeric value giving the size of points to plot (defaults to 2)
verbose	Boolean specifying if status messages should be reported.
plotALayers	list of ggplot2 layers to be passed to the rainfall plot.
plotBLayers	list of ggplot2 layers to be passed to the density plot.

### Slots

PlotA gtable object for the rainfall plot  
 PlotB gtable object for density plots based on the rainfall plot  
 Grob gtable object for the arranged plot  
 primaryData data.table object storing the primary data used for plotting.

---

SNPLoci *Identity snps*

---

**Description**

A data set containing locations of 24 identity snps originating from: Pengelly et al. Genome Med. 2013, PMID 24070238

**Usage**

```
data(SNPLoci)
```

**Format**

a data frame with 24 observations and 3 variables

**Value**

Object of class data frame

---

TvTi *Construct transition-transversion plot*

---

**Description**

Given a data frame construct a plot displaying the proportion or frequency of transition and transversion types observed in a cohort.

**Usage**

```
TvTi(
  x,
  fileType = NULL,
  y = NULL,
  clinData = NULL,
  type = "Proportion",
  lab_Xaxis = TRUE,
  lab_txtAngle = 45,
  palette = c("#D53E4F", "#FC8D59", "#FEE08B", "#E6F598", "#99D594", "#3288BD"),
  tvtiLayer = NULL,
  expeclayer = NULL,
  sort = "none",
  clinLegCol = NULL,
  clinVarCol = NULL,
  clinVarOrder = NULL,
  clinLayer = NULL,
  progress = TRUE,
  out = "plot",
  sample_order_input,
  layers = NULL,
  return_plot = FALSE
)
```

**Arguments**

x	Object of class data frame with rows representing transitions and transversions. The data frame must contain the following columns 'sample', 'reference' and 'variant' or alternatively "Tumor_Sample_Barcode", "Reference_Allele", "Tumor_Seq_Allele1", "Tumor_Seq_Allele2" depending on the argument supplied to the fileType parameter. (required)
fileType	Character string specifying the format the input given to parameter x is in, one of 'MAF', 'MGI'. The former option requires the data frame given to x to contain the following column names "Tumor_Sample_Barcode", "Reference_Allele", "Tumor_Seq_Allele1", "Tumor_Seq_Allele2" the later option requires the data frame given to x to contain the following column names "reference", "variant" and "sample". (required)
y	Named vector or data frame representing the expected transition and transversion rates. Either option must name transition and transversions as follows: "A->C or T->G (TV)", "A->G or T->C (TI)", "A->T or T->A (TV)", "G->A or C->T (TI)", "G->C or C->G (TV)", "G->T or C->A (TV)". If specifying a data frame, the data frame must contain the following columns names "Prop", "trans_tranv" (optional see vignette).
clinData	Object of class data frame with rows representing clinical data. The data frame should be in "long format" and columns must be names as "sample", "variable", and "value" (optional see details and vignette).
type	Character string specifying if the plot should display the Proportion or Frequency of transitions/transversions observed. One of "Proportion" or "Frequency", defaults to "Proportion".
lab_Xaxis	Boolean specifying whether to label the x-axis in the plot.
lab_txtAngle	Integer specifying the angle of labels on the x-axis of the plot.
palette	Character vector of length 6 specifying colours for each of the six possible transition transversion types.
tvtiLayer	Valid ggplot2 layer to be added to the main plot.
expecLayer	Valid ggplot2 layer to be added to the expected sub-plot.
sort	Character string specifying the sort order of the sample variables in the plot. Arguments to this parameter should be "sample", "tvti", or "none" to sort the x-axis by sample name, transition transversion frequency, or no sort respectively.
clinLegCol	Integer specifying the number of columns in the legend for the clinical data, only valid if argument is supplied to parameter clinData.
clinVarCol	Named character vector specifying the mapping of colours to variables in the variable column of the data frame supplied to clinData (ex. "variable"="colour").
clinVarOrder	Character vector specifying the order in which to plot variables in the variable column of the argument given to the parameter clinData. The argument supplied to this parameter should have the same unique length and values as in the variable column of the argument supplied to parameter clinData (see vignette).
clinLayer	Valid ggplot2 layer to be added to the clinical sub-plot.
progress	Boolean specifying if progress bar should be displayed for the function.
out	Character vector specifying the the object to output, one of "data", "grob", or "plot", defaults to "plot" (see returns).
sample_order_input	Sample orders to be used
layers	ggplot object to be added to proportions plot
return_plot	Return as ggplot object? Only returns main plot

**Details**

TvTi is a function designed to display proportion or frequency of transitions and transversion seen in a data frame supplied to parameter x.

**Value**

One of the following, a list of dataframes containing data to be plotted, a grob object, or a plot.

**Examples**

```
TvTi(brcaMAF, type='Frequency',
palette=c("#77C55D", "#A461B4", "#C1524B", "#93B5BB", "#4F433F", "#BFA753"),
lab_txtAngle=60, fileType="MAF")
```

---

VEP-class	<i>Class VEP</i>
-----------	------------------

---

**Description**

An S4 class for Variant Effect Predictor input, under development!!!

**Usage**

```
VEP(path, data = NULL, version = "auto", verbose = FALSE)
```

**Arguments**

path	String specifying the path to a VEP annotation file. Can accept wildcards if multiple VEP annotation files exist (see details).
data	data.table object storing a GMS annotation file. Overrides "path" if specified.
version	String specifying the version of the VEP files, Defaults to auto which will look for the version in the header.
verbose	Boolean specifying if progress should be reported while reading in the VEP files.

**Details**

When specifying a path to a VEP annotation file the option exist to either specify the full path to an annotation file or to use wildcards to specify multiple files. When specifying a full path the initializer will check if a column named "sample" containing the relevant sample for each row exists. If such a column is not found the initializer will assume this file corresponds to only one sample and populate a sample column accordingly. Alternatively if multiple files are specified at once using a wildcard, the initializer will aggregate all the files and use the file names minus any extension to populate sample names.

**Slots**

path Character string specifying the paths of the VEP files read in.  
version Numeric value specifying the version of VEP used.  
vepObject vep object which inherits from VEP\_Virtual class.

**See Also**[Waterfall](#)[MutSpectra](#)


---

 VEP\_v88-class                      *Class VEP\_v88*


---

**Description**

An S4 class to represent data in variant effect predictor version 88 format, inherits from the VEP\_Virtual class, under development!!!

**Usage**

```
VEP_v88(vepData, vepHeader)
```

**Arguments**

vepData	data.table object containing a VEP annotation file conforming to the version 88 specifications.
vepHeader	Object of class list containing character vectors for vep header information.

**Slots**

header	data.table object containing header information
description	data.table object containing column descriptions
position	data.table object containing column names "chromosome_name", "start", "stop".
mutation	data.table object containing column names "reference", "variant", "trv_type".
sample	data.table object containing columns names "sample".
meta	data.table object containing meta data.

---

 VEP\_Virtual-class                      *Class VEP\_Virtual*


---

**Description**

An S4 class to act as a virtual class for VEP version sub-classes, under development!!!

**Slots**

header	data.table object holding header information.
description	data.table object holding column descriptions
position	data.table object holding genomic positions.
mutation	data.table object holding mutation status data.
sample	data.table object holding sample data.
meta	data.table object holding all other meta data.

---

waterfall	<i>Construct a oncoprint</i>
-----------	------------------------------

---

**Description**

This function has been removed, please use Waterfall() (capital W) Tutorial can be found at: <https://currentprotocols.onlinelibrary.wiley.com/doi/full/10.1002/cpz1.252>

**Usage**

```
waterfall()
```

---

Waterfall-class	<i>Class Waterfall</i>
-----------------	------------------------

---

**Description**

An S4 class for the waterfall plot object, under development!!!

**Usage**

```
Waterfall(
  input,
  labelColumn = NULL,
  samples = NULL,
  coverage = NULL,
  mutation = NULL,
  genes = NULL,
  mutationHierarchy = NULL,
  recurrence = NULL,
  geneOrder = NULL,
  geneMax = NULL,
  sampleOrder = NULL,
  plotA = c("frequency", "burden", NULL),
  plotATally = c("simple", "complex"),
  plotALayers = NULL,
  plotB = c("proportion", "frequency", NULL),
  plotBTally = c("simple", "complex"),
  plotBLayers = NULL,
  gridOverlay = FALSE,
  drop = TRUE,
  labelSize = 5,
  labelAngle = 0,
  sampleNames = TRUE,
  clinical = NULL,
  sectionHeights = NULL,
  sectionWidths = NULL,
  verbose = FALSE,
  plotCLayers = NULL
)
```

**Arguments**

input	Object of class <code>MutationAnnotationFormat</code> , <code>VEP</code> , <code>GMS</code> , or alternatively a data frame/data table with column names "sample", "gene", "mutation".
labelColumn	Character vector specifying a column name from which to extract label names for cells, must be a column within the object passed to input.
samples	Character vector specifying samples to plot. If not NULL all samples in "input" not specified with this parameter are removed. Further samples specified but not present in the data will be added.
coverage	Integer specifying the size in base pairs of the genome covered by sequence data from which mutations could be called. Required for the mutation burden sub-plot (see details and vignette). Optionally a named vector of integers corresponding to each sample can be supplied for more accurate calculations.
mutation	Character vector specifying mutations to keep, if defined mutations not supplied are removed from the main plot.
genes	Character vector specifying genes to keep, if not "NULL" all genes not specified are removed. Further genes specified but not present in the data will be added.
mutationHierarchy	data.table/data.frame object with rows specifying the order of mutations from most to least deleterious and containing column names "mutation" and "color". Used to change the default colors and/or to give priority to a mutation for the same gene/sample (see details and vignette).
recurrence	Numeric value between 0 and 1 specifying a mutation recurrence cutoff. Genes which do not have mutations in the proportion of samples defined are removed.
geneOrder	Character vector specifying the order in which to plot genes.
geneMax	Integer specifying the maximum number of genes to be plotted. Genes kept will be chosen based on the recurrence of mutations in samples, unless geneOrder is specified.
sampleOrder	Character vector specifying the order in which to plot samples.
plotA	String specifying the type of plot for the top sub-plot, one of "burden", "frequency", or NULL for a mutation burden (requires coverage to be specified), frequency of mutations, or no plot respectively.
plotATally	String specifying one of "simple" or "complex" for a simplified or complex tally of mutations respectively.
plotALayers	list of ggplot2 layers to be passed to the plot.
plotB	String specifying the type of plot for the left sub-plot, one of "proportion", "frequency", or NULL for a plot of gene proportions frequencies, or no plot respectively.
plotBTally	String specifying one of "simple" or "complex" for a simplified or complex tally of genes respectively.
plotBLayers	list of ggplot2 layers to be passed to the plot.
gridOverlay	Boolean specifying if a grid should be overlaid on the waterfall plot. This is not recommended for large cohorts.
drop	Boolean specifying if mutations not in the main plot should be dropped from the legend. If FALSE the legend will be based on mutations in the data before any subsets occur.
labelSize	Integer specifying the size of label text within each cell if "labelColumn" has been specified.

labelAngle	Numeric value specifying the angle of label text if "labelColumn" has been specified.
sampleNames	Boolean specifying if samples should be labeled on the x-axis of the plot.
clinical	Object of class <a href="#">Clinical</a> , used for adding a clinical data subplot.
sectionHeights	Numeric vector specifying relative heights of each plot section, should sum to one. Expects a value for each section.
sectionWidths	Numeric vector specifying relative heights of each plot section, should sum to one. Expects a value for each section.
verbose	Boolean specifying if status messages should be reported.
plotCLayers	list of ggplot2 layers to be passed to the main plot.

### Details

‘Waterfall()’ is designed to visualize the mutations seen in a cohort. As input the function takes an object of class [MutationAnnotationFormat](#), [VEP](#), or [GMS](#). Alternatively a user can provide either of `data.table` or `data.frame` as long as the column names of those objects include "sample", "gene", and "mutation". When supplying an object of class `data.table` or `data.frame` the user must also provide input to the ‘mutationHierarchy’ parameter.

The ‘mutationHierarchy’ parameter expects either a `data.table` or `data.frame` object containing the column names "mutation" and "color". Each row should match a mutation type given in the param ‘input’. The ‘mutationHierarchy’ parameter is intended to both change the colors of mutations on the plot and to set a hierarchy of which mutation type to plot if there are more than 1 mutation types for the same gene/sample combination.

### Slots

PlotA	<code>gtable</code> object for the top sub-plot.
PlotB	<code>gtable</code> object for the left sub-plot.
PlotC	<code>gtable</code> object for the main plot.
PlotD	<code>gtable</code> object for the bottom sub-plot.
Grob	<code>gtable</code> object for the arranged plot.
primaryData	<code>data.table</code> object storing the primary data, should have column names sample, gene, mutation, label.
simpleMutationCounts	<code>data.table</code> object storing simplified mutation counts, should have column names sample, mutation, Freq, mutationBurden
complexMutationCounts	<code>data.table</code> object storing mutation counts per mutation type should have column names sample, mutation, Freq, mutationBurden.
geneData	<code>data.table</code> object storing gene counts, should have column names gene, mutation, count.
ClinicalData	<code>data.table</code> object storing the data used to plot the clinical sub-plot.
mutationHierarchy	<code>data.table</code> object storing the hierarchy of mutation type in order of most to least important and the mapping of mutation type to color. Should have column names mutation, color, and label.

### See Also

[MutationAnnotationFormat](#), [VEP](#), [GMS](#), [Clinical](#)



**Examples**

```

set.seed(426)

# create a data frame with required column names
mutationDF <- data.frame("sample"=sample(c("sample_1", "sample_2", "sample_3"), 10, replace=TRUE),
                        "gene"=sample(c("egfr", "tp53", "rb1", "apc"), 10, replace=TRUE),
                        "mutation"=sample(c("missense", "frame_shift", "splice_site"), 10, replace=TRUE))

# set the mutation hierarchy (required for DF)
hierarchyDF <- data.frame("mutation"=c("missense", "frame_shift", "slice_site"),
                        "color"=c("#3B3B98", "#BDC581", "#6A006A"))

# Run the Waterfall Plot and draw the output
Waterfall.out <- Waterfall(mutationDF, mutationHierarchy=hierarchyDF)
drawPlot(Waterfall.out)

```

writeData

*Method writeData***Description**

Method writeData

**Usage**

```

writeData(object, ...)

## S4 method for signature 'GMS_Virtual'
writeData(object, file, sep, ...)

## S4 method for signature 'GMS'
writeData(object, file, ...)

## S4 method for signature 'MutationAnnotationFormat_Virtual'
writeData(object, file, sep, ...)

## S4 method for signature 'MutationAnnotationFormat'
writeData(object, file, ...)

## S4 method for signature 'VEP_Virtual'
writeData(object, file, sep, ...)

## S4 method for signature 'VEP'
writeData(object, file, ...)

```

**Arguments**

object	Object of class VEP
...	additional arguments to passed
file	Character string specifying a file to send output to.
sep	Delimiter used when writing output, defaults to tab.

**Details**

The `writeData` method is used to output data held in GenVisR objects to a file.

# Index

## \* datasets

- brcaMAF, 4
- cytoGeno, 12
- HCC1395\_Germline, 26
- HCC1395\_N, 27
- HCC1395\_T, 27
- hg19chr, 28
- LucCNseg, 34
- PIK3CA, 40
- SNPloci, 42
- .getData\_Lollipop (getData), 17
- .getData\_MutSpectra (getData), 17
- .getData\_Rainfall (getData), 17
- .getData\_waterfall (getData), 17
  
- brcaMAF, 4
  
- Clinical, 48
- Clinical (Clinical-class), 4
- Clinical-class, 4
- cnFreq, 5
- cnSpec, 7
- cnView, 8
- compIdent, 10
- covBars, 11
- cytoGeno, 12
  
- drawPlot, 5, 12
- drawPlot, Clinical-method (drawPlot), 12
- drawPlot, Lollipop-method (drawPlot), 12
- drawPlot, MutSpectra-method (drawPlot), 12
- drawPlot, Rainfall-method (drawPlot), 12
- drawPlot, Waterfall-method (drawPlot), 12
  
- genCov, 13
- geneViz, 15
- GenVisR, 17
- getData, 5, 17
- getData, Clinical-method (getData), 17
- getData, ClinicalData-method (getData), 17
- getData, Lollipop-method (getData), 17
- getData, LollipopData-method (getData), 17
  
- getData, MutSpectra-method (getData), 17
- getData, MutSpectraPrimaryData-method (getData), 17
- getData, Rainfall-method (getData), 17
- getData, RainfallPrimaryData-method (getData), 17
- getData, Waterfall-method (getData), 17
- getData, WaterfallData-method (getData), 17
  
- getDescription, 19
- getDescription, VEP-method (getDescription), 19
- getDescription, VEP\_Virtual-method (getDescription), 19
  
- getGrob, 19
- getGrob, Lollipop-method (getGrob), 19
- getGrob, LollipopPlots-method (getGrob), 19
- getGrob, MutSpectra-method (getGrob), 19
- getGrob, MutSpectraPlots-method (getGrob), 19
- getGrob, Rainfall-method (getGrob), 19
- getGrob, RainfallPlots-method (getGrob), 19
- getGrob, Waterfall-method (getGrob), 19
- getGrob, WaterfallPlots-method (getGrob), 19
  
- getHeader, 20
- getHeader, VEP-method (getHeader), 20
- getHeader, VEP\_Virtual-method (getHeader), 20
  
- getMeta, 20
- getMeta, GMS-method (getMeta), 20
- getMeta, GMS\_Virtual-method (getMeta), 20
- getMeta, MutationAnnotationFormat-method (getMeta), 20
- getMeta, MutationAnnotationFormat\_Virtual-method (getMeta), 20
- getMeta, VEP-method (getMeta), 20
- getMeta, VEP\_Virtual-method (getMeta), 20
  
- getMutation, 21
- getMutation, GMS-method (getMutation), 21
- getMutation, GMS\_Virtual-method

- (getMutation), 21
- getMutation, MutationAnnotationFormat-method (getMutation), 21
- getMutation, MutationAnnotationFormat\_Virtual-method (getMutation), 21
- getMutation, VEP-method (getMutation), 21
- getMutation, VEP\_Virtual-method (getMutation), 21
- getPath, 22
- getPath, GMS-method (getPath), 22
- getPath, MutationAnnotationFormat-method (getPath), 22
- getPath, VEP-method (getPath), 22
- getPosition, 22
- getPosition, GMS-method (getPosition), 22
- getPosition, GMS\_Virtual-method (getPosition), 22
- getPosition, MutationAnnotationFormat-method (getPosition), 22
- getPosition, MutationAnnotationFormat\_Virtual-method (getPosition), 22
- getPosition, VEP-method (getPosition), 22
- getPosition, VEP\_Virtual-method (getPosition), 22
- getSample, 23
- getSample, GMS-method (getSample), 23
- getSample, GMS\_Virtual-method (getSample), 23
- getSample, MutationAnnotationFormat-method (getSample), 23
- getSample, MutationAnnotationFormat\_Virtual-method (getSample), 23
- getSample, VEP-method (getSample), 23
- getSample, VEP\_Virtual-method (getSample), 23
- getVersion, 24
- getVersion, GMS-method (getVersion), 24
- getVersion, MutationAnnotationFormat-method (getVersion), 24
- getVersion, VEP-method (getVersion), 24
- GMS, 47, 48
- GMS (GMS-class), 24
- GMS-class, 24
- GMS\_v4 (GMS\_v4-class), 25
- GMS\_v4-class, 25
- GMS\_Virtual-class, 26
- HCC1395\_Germline, 26
- HCC1395\_N, 27
- HCC1395\_T, 27
- hg19chr, 28
- ideoView, 28
- lohSpec, 29
- lohView, 31
- Lollipop (Lollipop-class), 32
- Lollipop, 32
- Lollipop-class, 32
- LucCNseg, 34
- MutationAnnotationFormat, 47, 48
- MutationAnnotationFormat (MutationAnnotationFormat-class), 34
- MutationAnnotationFormat-class, 34
- MutationAnnotationFormat\_v1.0 (MutationAnnotationFormat\_v1.0-class), 35
- MutationAnnotationFormat\_v1.0-class, 35
- MutationAnnotationFormat\_v2.0 (MutationAnnotationFormat\_v2.0-class), 36
- MutationAnnotationFormat\_v2.0-class, 36
- MutationAnnotationFormat\_v2.1 (MutationAnnotationFormat\_v2.1-class), 36
- MutationAnnotationFormat\_v2.1-class, 36
- MutationAnnotationFormat\_v2.2 (MutationAnnotationFormat\_v2.2-class), 37
- MutationAnnotationFormat\_v2.2-class, 37
- MutationAnnotationFormat\_v2.3 (MutationAnnotationFormat\_v2.3-class), 37
- MutationAnnotationFormat\_v2.3-class, 37
- MutationAnnotationFormat\_v2.4 (MutationAnnotationFormat\_v2.4-class), 38
- MutationAnnotationFormat\_v2.4-class, 38
- MutationAnnotationFormat\_Virtual-class, 39
- MutSpectra, 25, 35, 45
- MutSpectra (MutSpectra-class), 39
- MutSpectra-class, 39
- PIK3CA, 40
- Rainfall (Rainfall-class), 41
- Rainfall-class, 41
- SNPloci, 42

TvTi, [42](#)

VEP, [47](#), [48](#)

VEP (VEP-class), [44](#)

VEP-class, [44](#)

VEP\_v88 (VEP\_v88-class), [45](#)

VEP\_v88-class, [45](#)

VEP\_Virtual-class, [45](#)

Waterfall, [25](#), [35](#), [45](#)

Waterfall (Waterfall-class), [46](#)

waterfall, [46](#)

Waterfall-class, [46](#)

writeData, [49](#)

writeData,GMS-method (writeData), [49](#)

writeData,GMS\_Virtual (writeData), [49](#)

writeData,GMS\_Virtual-method  
(writeData), [49](#)

writeData,MutationAnnotationFormat-method  
(writeData), [49](#)

writeData,MutationAnnotationFormat\_Virtual  
(writeData), [49](#)

writeData,MutationAnnotationFormat\_Virtual-method  
(writeData), [49](#)

writeData,VEP-method (writeData), [49](#)

writeData,VEP\_Virtual (writeData), [49](#)

writeData,VEP\_Virtual-method  
(writeData), [49](#)