Package 'GenVisR'

October 12, 2016

Title Genomic Visualizations in R

Version 1.0.4

Maintainer Zachary Skidmore <zskidmor@genome.wustl.edu>

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

Depends R (>= 3.3.0)

Imports AnnotationDbi, biomaRt, BiocGenerics, Biostrings, DBI, FField, GenomicFeatures, GenomicRanges, ggplot2 (>= 0.9.2), grid, gridExtra, gtable, gtools, IRanges, plyr (>= 1.8.3), reshape2, Rsamtools, scales, stats, utils, viridis

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

VignetteBuilder knitr

RoxygenNote 5.0.1

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]

R topics documented:

`brcaMAF` *Truncated BRCA MAF file*

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#TCGAMAFFiles-BRCA:Breastinvasivecarcinoma, /dccfiles_prod/tcgafiles/distro_ftpusers/anonymous/tumor/brca/gsc/genome.wustl.edu/illur

Usage

```
data(brcaMAF)
```

Format

a data frame with 2773 observations and 55 variables

Value

Object of class data drame

`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, out = "plot")
```

Arguments

x	Object of class data frame with rows representing the proportion of CN losses/gains across the genome (default), or actual CN values. The former option must contain columns with the following names "chromosome", "start", "end", "gain", and "loss", and the latter option must contain column names "chromosome", "start", "end", "segmean", and "sample". Windows supplied must be consistent across samples!
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.
out	Character vector specifying the the object to output, one of "data", "grob", or "plot", defaults to "plot" (see returns).

Details

cnFreq will detect the column names present in the data frame supplied to x, and will perform one of the following actions. If "gain" and "loss" columns are detected the raw data will be plotted, if "segmean" and "sample" columns are detected the frequency of copy-number gains and losses present in the cohort will be calculated and plotted. 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

```
# Create data
xstart <- seq(0,4990000,length.out=500)
xloss <- rep(runif(10,0,0.6),rep(50,10))/1.5
xloss <- xloss + jitter(xloss,amount=0.002)
```

```
x <- data.frame(chromosome=rep(paste0("chr",1:5),rep(500,5)), start=xstart,
end=xstart+10000, loss=xloss, gain=(1-xloss))

# Plot the data
cnFreq(x)
```

`cnFreq_buildMain` *Construct CN frequency plot*

Description

given a data frame construct a plot to display proportions of losses and gains across the genome

Usage

```
cnFreq_buildMain(data_frame, plotType, plot_title = NULL,
  CN_low_colour = "#002EB8", CN_high_colour = "#A30000", x_lab_size = 12,
  y_lab_size = 12, facet_lab_size = 10, plotLayer = NULL)
```

Arguments

`data_frame`	object of class data frame containing columns chromosome, start, end, gain, and loss
`plotType`	character string to determine whether to plot proportions or frequencies
`plot_title`	character string for title of plot
`CN_low_colour`	character string specifying low value of colour gradient
`CN_high_colour`	character string specifying high value of colour gradient
`x_lab_size`	integer specifying the size of the X label
`y_lab_size`	integer specifying the size of the Y label
`facet_lab_size`	integer specifying the size of the faceted labels
`plotLayer`	Additional layer to be plotted, can be a theme but must be a ggplot layer

Value

ggplot object

cnFreq_qual *check input to cnFreq*

Description

Perform a data quality check for inputs to cnFreq

Usage

```
cnFreq_qual(x)
```

Arguments

x	a data frame with columns chromosome, start, end, gain, and loss, or chromosome, start, end, segmean, and sample

Value

list containing data frame passing quality checks and the type of plot (proportional losses/gain or frequency of losses/gains)

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

x	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".
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).
plot_title	Character string specifying title to display on the plot.

CN_Loss_colour	Character string specifying the colour value of copy number losses.
CN_Gain_colour	Character string specifying the colour value of 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.
out	Character vector specifying the the object to output, one of "data", "grob", or "plot", defaults to "plot" (see returns).
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"

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 desireable to plot a specific region encapsulating the copy number data these boundaries can be supplied to the ‘y‘ paramter 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")
```

---

cnSpec_buildMain *Construct CN cohort plot*

Description

given a data frame construct a plot to display CN information for a group of samples

Usage

```
cnSpec_buildMain(data_frame, dummy_data, plot_title = NULL,
  CN_low_colour = "#002EB8", CN_high_colour = "#A30000", x_lab_size = 12,
  y_lab_size = 12, facet_lab_size = 10, layers = NULL,
  CNscale = "absolute")
```

Arguments

data_frame	object of class data frame containing columns chromosome, start, end, cn, sample
dummy_data	Object of class data frame containing columns chromosome, start, end, cn, sample. Used for defining chromosome boundaries
plot_title	character string for title of plot
CN_low_colour	character string specifying low value of colour gradient
CN_high_colour	character string specifying high value of colour gradient
x_lab_size	integer specifying the size of the X label
y_lab_size	integer specifying the size of the Y label
facet_lab_size	integer specifying the size of the faceted labels
layers	Additional layers to be plotted, can be a theme but must be a ggplot layer
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"

Value

ggplot object

---

cnSpec_qual *Construct CN cohort plot*

Description

given a data frame construct a plot to display CN information for a group of samples

Usage

```
cnSpec_qual(x, y, genome, CNscale)
```

Arguments

x	object of class data frame containing columns chromosome, start, stop, segmean, sample
y	object of class data frame containing user supplied chromosome locations
genome	character string specifying a user supplied genome
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"

Value

character string specifying input passed quality check

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")
```

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).

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)
```

---

cnView_buildMain *construct CN plot*

Description

given a CN data frame plot points in ggplot

Usage

```
cnView_buildMain(x, y, z = NULL, chr, CNscale = FALSE, layers = NULL)
```

Arguments

x	a data frame with columns chromosome, coordinate, cn, p_value
y	a data frame with columns chromosome, coordinate for plotting boundaries
z	a data frame with columns chromsome, start, end, segmean specifying segments called from copy number (optional)
chr	a character string specifying chromosome
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"
layers	additional ggplot2 layers to add

Value

ggplot2 object

`cnView_qual` *check input to cnView*

Description

Perform a data quality check for inputs to cnView

Usage

```
cnView_qual(x, y, z, genome, CNscale)
```

Arguments

`x`	a data frame with columns chromosome, coordinate, cn
`y`	a data frame with columns "chrom", "chromStart", "chromEnd", "name", "gieStain"
`z`	a data frame with columns chromosome, start, end , segmean
`genome`	character string specifying UCSC genome to use
`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"

Value

a list of data frames passing quality checks

`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 comppare 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 paramter. 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 afore mentioned 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)
```

compIdent_bamRcnt *Count nucleotide reads at SNP locations*

Description

Given the bam file path, count the number of reads at specified snp locations

Usage

```
compIdent_bamRcnt(bamfile, genome, target = NULL, debug = FALSE)
```

Arguments

bamfile	Path to the bam file
genome	Object of class BSgenome corresponding to a genome of interest
target	Object of class data frame containing target locations in 1-base format and containing columns "chr", "start", "end", "var", "name"
debug	Boolean specifying if test datasets should be used for debugging.

Value

object of class data frame containing readcount information

compIdent_bamRcnt_qual

Count nucleotide reads at SNP locations

Description

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

Usage

```
compIdent_bamRcnt_qual(genome, targetbed)
```

Arguments

genome	Object of class BSgenome corresponding to a genome of interest
targetbed	Object of class data frame containing target locations in .bed format

Value

list of data objects passing quality checks

compIdent_buildMain *Compare sample identities*

Description

Produce an identity SNP plot displaying VAFs of 24 SNP locations and coverage information to compare multiple sample identities

Usage

```
compIdent_buildMain(x, mainLayer = NULL, covLayer = NULL)
```

Arguments

x	Data frame of vaf for each sample
mainLayer	Valid ggplot2 layer for altering the main plot.
covLayer	Valid ggplot2 layer for altering the coverage plot.

Value

ggplot2 grob object

compIdent_format *Format readcount tables from compIdent*

Description

Format readcount tables from compIdent for input into compIdent_buildMain

Usage

```
compIdent_format(x)
```

Arguments

x	Named list of data frames with rows of the data frame corresponding to target locations.

Value

Formated data frame

covBars *Construct an overall coverage cohort plot*

Description

Given a matrix construct a plot to display sequencing depth acheived 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

x	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
colour	Character vector specifying colours to represent sequencing depth.
plot_title	Character string specifying the title to display on the plot.
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.
out	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)
```

covBars_buildMain *Construct coverage cohort plot*

Description

given a data frame construct a plot to display coverage as percentage bars for a group of samples

Usage

```
covBars_buildMain(data_frame, col, plot_title = NULL, x_lab_size = 12,
  y_lab_size = 12, facet_lab_size = 10, layers = NULL)
```

Arguments

data_frame	object of class data frame containing columns depth, sample, bp
col	vector of colors for the coverage bars
plot_title	character string for title of plot
x_lab_size	integer specifying the size of the X label
y_lab_size	integer specifying the size of the Y label
facet_lab_size	integer specifying the size of the faceted labels
layers	Additional layers to be plotted, can be a theme but must be a ggplot layer

Value

ggplot object

covBars_qual *Construct coverage cohort plot*

Description

given a matrix construct a plot to display coverage as percentage bars for a group of samples

Usage

```
covBars_qual(x)
```

Arguments

x	object of class matrix containing rows for the coverage and columns the sample names

Value

a list of data frame and color vector

cytoGeno *Cytogenetic banding dataset*

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

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 rack 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)
```

genCov_alignPlot *align plots on an axis*

Description

given a list of plots, align them on plotting space

Usage

```
genCov_alignPlot(plot_list, axis = "both")
```

Arguments

plot_list	list of ggplot objects
axis	character string to specify the axis to align plotting space on, one of both, width, height

Value

ggplotGrob object

genCov_assign_ggplotGrob_height *assign ggplotGrob height*

Description

assign height of ggplotGrob object

Usage

```
genCov_assign_ggplotGrob_height(x, max_height)
```

Arguments

x	ggplotGrob object
max_height	grob object specifying width to reassign ggplotGrob with

Value

ggplotGrob

genCov_assign_ggplotGrob_width *assign ggplotGrob width*

Description

assign width of ggplotGrob object

Usage

```
genCov_assign_ggplotGrob_width(x, max_width)
```

Arguments

x	ggplotGrob object
max_width	grob object specifying width to reassign ggplotGrob with

Value

ggplotGrob

genCov_buildCov *build coverage plot*

Description

given data build a coverage plot to represent the data

Usage

```
genCov_buildCov(data_frame, x_limits = NULL, display_x_axis = TRUE,
  colour = "blue", plot_type = "point", layers = NULL)
```

Arguments

data_frame	an object of class data frame containing columns stop and cov
x_limits	vector giving x-axis limits for plot, inferred from data if not specified
display_x_axis	boolean specifying whether to plot x-axis labels
colour	character string specifying the color of the data in the plot
plot_type	character string specifying one of line, bar, or point for data display
layers	additional ggplot2 layers to plot

Value

ggplot object

genCov_buildTrack *build label for plot*

Description

given a name create a label

Usage

```
genCov_buildTrack(name, bg_fill = "black", text_fill = "white",
  border = "black", size = 10)
```

Arguments

name	character string giving the name of the track
bg_fill	character string giving the colour to fill the label
text_fill	character string giving the colour to fill the text
border	character string specifying the colour to fill the border of the label
size	integer specifying the size of the text within the label

Value

ggplot object

genCov_extr_ggplotGrob_height *extract ggplotGrob height*

Description

extract plot height of ggplotGrob object

Usage

```
genCov_extr_ggplotGrob_height(x)
```

Arguments

x	ggplotGrob object

Value

ggplotGrob height parameters

genCov_extr_ggplotGrob_width *extract ggplotGrob width*

Description

extract plot width of ggplotGrob object

Usage

```
genCov_extr_ggplotGrob_width(x)
```

Arguments

x	ggplotGrob object

Value

ggplotGrob width parameters

genCov_qual *Perform quality control on genCov data*

Description

Ensure data input into genCov is of the proper type and format

Usage

```
genCov_qual(x = x, txdb = txdb, gr = gr, genome = genome)
```

Arguments

x	named list containing data frames with columns end and cov
txdb	A TxDb object for a genome
gr	A Granges object specifying a region of interest
genome	Object of class BSgenome specifying the genome

Value

a list of objects passing basic quality control

genCov_trackViz *Overlay tracks with plots*

Description

given a named list of plots, display them on tracks

Usage

```
genCov_trackViz(..., bgFill = "black", textFill = "white",
  border = "black", size = 10, axis_align = "none", widthRatio = c(1,
  10), list = TRUE)
```

Arguments

...	named list of ggplot2 plots
bgFill	character string giving the colour to fill the label
textFill	character string giving the colour to fill the text
border	character string specifying the colour to fill the border of the label
size	integer specifying the size of the text within the label
axis_align	character string specifying axis to align plotting space on, one of 'both', 'height', 'width', 'none'
widthRatio	vector of length 2 giving the ratio of track labels to plots
list	boolean specifying whether plots are in a named list or specified individually via ...

Value

ggplotGrob object

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 that 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)
```

`geneViz_buildGene` *build gene plot*

Description

given a data frame with gene feature information build the ggplot2 object

Usage

```
geneViz_buildGene(data_frame, display_x_axis = TRUE, x_limits = NULL,
  gene_colour = NULL, transcript_name = FALSE, transcript_name_size = 4,
  layers = NULL)
```

Arguments

`data_frame`	an object of class data frame specifying gene feature information
`display_x_axis`	Boolean specifying whether to display X axis coordinate values
`x_limits`	vector specifying x-axis limits of plot
`gene_colour`	character specifying colour of gene to be plotted
`transcript_name`	Boolean specifying whether to plot USCS transcript names
`transcript_name_size`	Integer specifying the size of the transcript name text
`layers`	additional ggplot2 layers to plot

Value

ggplot object

geneViz_calcGC *Calculate GC content*

Description

Calculate GC content for elements in a GRanges object

Usage

```
geneViz_calcGC(gr, genome)
```

Arguments

gr	A Granges object to calculate GC content for
genome	Object of class BSgenome specifying the genome to calculate GC content from

Value

Object of class GRanges

geneViz_cdsFromTXID *cdsFromTXID*

Description

Return CDS coordinates as a GRanges object given transcript IDs

Usage

```
geneViz_cdsFromTXID(txdb, txid)
```

Arguments

txdb	A TxDb object for a genome
txid	A list of TXIDs

Value

Object of class Granges

geneViz_extrCDS *Extract CDS*

Description

Extract CDS coordinates within a GRanges object given a transcription database

Usage

```
geneViz_extrCDS(txdb = NULL, gr = NULL, reduce = FALSE, gaps = FALSE)
```

Arguments

txdb	A TxDb object for a genome
gr	A Granges object specifying the region of interest
reduce	Boolean specifying whether to collapse isoforms
gaps	Boolean specifying whether to report space between CDS instead of CDS

Value

Object of class Granges list

geneViz_extrUTR *Extract UTR*

Description

Extract UTR coordinates within a GRanges object given a transcription database

Usage

```
geneViz_extrUTR(txdb = txdb, gr = gr, reduce = FALSE, gaps = FALSE)
```

Arguments

txdb	A TxDb object for a genome
gr	A Granges object specifying the region of interest
reduce	Boolean specifying whether to collapse isoforms
gaps	Boolean specifying whether to report space between UTR instead of UTR

Value

Object of class Granges list

geneViz_formatCDS *format cds*

Description

given a Granges object specifying a region of interest, format into a form recognizable by ggplot2

Usage

```
geneViz_formatCDS(txdb = NULL, gr = NULL, genome = NULL, reduce = FALSE)
```

Arguments

txdb	A TxDb object for a genome
gr	A Granges object to format
genome	Object of class BSgenome specifying the genome for GC content calculation
reduce	Boolean specifying whether to collapse isoforms in the Granges object ROI

Value

Object of class data frame

geneViz_formatUTR *format UTR*

Description

given a Granges object specifying a region of interest, format into a form recognizable by ggplot2

Usage

```
geneViz_formatUTR(txdb = NULL, gr = NULL, genome = NULL, reduce = FALSE)
```

Arguments

txdb	A TxDb object for a genome
gr	A Granges object to format
genome	Object of class BSgenome specifying the genome for GC content calculation
reduce	Boolean specifying whether to collapse isoforms in the Granges object ROI

Value

Object of class data frame

geneViz_Granges2dataframe *Convert Granges object to dataframe*

Description

Convert a Granges object with meta data GC content to a object of class data frame

Usage

```
geneViz_Granges2dataframe(gr)
```

Arguments

gr	A Granges object to convert to data frame

Value

Object of class data frame

geneViz_mapCoordSpace *Map regions to transformed space*

Description

Reference a master genomic region file to map original positions to a transformed space

Usage

```
geneViz_mapCoordSpace(master, coord)
```

Arguments

master	an object of class data frame containing columns start, end, width, type, trans_start, trans_end representing a master genomic region with features from isoforms merged
coord	an object of class data frame containing columns start and end to map to transformed space

Value

an object of class data frame identical to coord but with extra columns for transformed coord

geneViz_mapCovCoordSpace *Map coverage track regions to transformed space*

Description

Reference a master genomic region file to map original positions of a coverage track to a transformed space

Usage

```
geneViz_mapCovCoordSpace(cov.coords, master)
```

Arguments

cov.coords	an object of class data frame containing columns start and end to map to transformed space, with rows demarking single nucleotide coverages
master	an object of class data frame containing columns start, end, width, type, trans_start, trans_end representing a master genomic region with features from isoforms merged

Value

an object of class data frame identical to coord but with extra columns for transformed coord

geneViz_mergeRegions *Create Region Table*

Description

Create a master region table by merging isoforms

Usage

```
geneViz_mergeRegions(gene_features, gr, base, transform)
```

Arguments

gene_features	A dataframe specifying features of a gene
gr	Granges object specifying the ROI
base	A vector of log bases to transform the data, corresponding to the elements of transform
transform	A vector of strings designating what objects to log transform

Value

Master region table data frame

geneViz_mergeTypeRegions *Create Typed Region Table*

Description

Create a master region table by merging isoforms

Usage

```
geneViz_mergeTypeRegions(type.master)
```

Arguments

type.master	A dataframe of all elements of a certain type, such as CDS

Value

type.master A dataframe of merged elements of a certain type

geneViz_mergeTypes *Merge Typed Region Tables*

Description

Create a master region table by merging isoforms

Usage

```
geneViz_mergeTypes(master)
```

Arguments

master	An unsorted dataframe of CDS and UTR elements before merging

Value

master A sorted dataframe of merged CDS and UTR elements

GenVisR *GenVisR*

Description

GenVisR

HCC1395_Germline *Germline Calls*

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 *hg19 chromosome boundaries*

Description

A data set containg 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 *Construct an ideogram*

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 inforamtion. 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)
```

ideoView_buildMain *build chromosome*

Description

given a data frame with cytogenetic band locations plot chromosome in ggplot

Usage

```
ideoView_buildMain(data_frame, chromosome, chr_txt_angle = chr_txt_angle,
  chr_txt_size = chr_txt_size, layers = NULL)
```

Arguments

data_frame	a data frame with columns chrom, chromStart, chromEnd, name, gieStain, height_min, height_max, alternate, bandcenter, text_y, arm
chromosome	character string specifying UCSC chromosome to plot one of chr... or all
chr_txt_angle	integer specifying angle of text when plotting band text
chr_txt_size	integer specifying size of text when plotting band text
layers	additional ggplot2 layers to plot

Value

ggplot object

ideoView_formatCytobands *reformat cytogenetic band data frame*

Description

given a data frame of cytogenetic bands, format it for ggplot call

Usage

```
ideoView_formatCytobands(data_frame, chromosome)
```

Arguments

data_frame	a data frame retrieved from UCSC giving cytogenetic information
chromosome	character string specifying chromosome of interest from UCSC data frame

Value

object of class data frame

ideoView_qual *Check input to ideoView*

Description

Check that input to ideoView is properly formatted

Usage

```
ideoView_qual(x)
```

Arguments

x	a data frame with rows representing cytogenetic bands for a genome. The data frame should have columns "chrom", "chromStart", "chromEnd", "name", "gieStain'.

Value

data frame

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).
path	Character string specifying the path to a directory containing germline calls for each sample. Germline calls are expected to be stored as tab-seperated 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 0-50 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 ggpot2 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 0-100. 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 requries 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)
```

`lohSpec_buildMain` *Plot LOH data*

Description

Build a ggplot2 object displaying calculated LOH data

Usage

```
lohSpec_buildMain(x, dummyData, colourScheme = "inferno", plotLayer = NULL)
```

Arguments

`x`	object of class dataframe with loh difference calculations and column names "window_start", "window_stop", "chromosome", "sample", and "loh_diff"
`dummyData`	object of class dataframe with column names "chromosome", "start", "end" specifying chromosome boundaries
`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. for the legend's parameters

Value

object of class ggplot2

`lohSpec_fileGlob` *Grab data for lohSpec*

Description

Look in the specified file path and grab data with the proper extension for lohSpec

Usage

```
lohSpec_fileGlob(path, fileExt, step, window_size, gender)
```

Arguments

path	character string specifying which directory contains the sample information stored as datasets with columns "chromosome", "position", "n_vaf", "t_vaf", and "sample" (required if x is not specified)
fileExt	character string specifying the file extensions of files
step	integer with the length of divisions (bp) in chromosomes
window_size	Integer value specifying the size of the window in base pairs in which to calculate the mean Loss of Heterozygosity.
gender	vector of length equal to the number of samples, consisting of elements from the set "M", "F"

Value

object of class data frame from data specified in path for lohSpec

---

lohSpec_lohCalc *Calculate loh difference*

Description

Obtain LOH on an entire chromsomes from samples in a cohort

Usage

```
lohSpec_lohCalc(window_data, out, normal)
```

Arguments

window_data	object of class data frame with columns 'window_start' and 'window_stop
out	object of class dataframe with columns 'chromosome', 'position', 'n_vaf', 't_vaf', and 'sample'
normal	integer specifying the subtraction value from tumor VAF

Value

object of class dataframe containing mean LOH difference calculations and column names "window_start", "window_stop", "chromosome", "position", "n_vaf", "t_vaf", "sample", "loh_diff"

lohSpec_qual *Check input to lohSpec*

Description

Perform data quality checks on input supplied to lohSpec

Usage

```
lohSpec_qual(x, y, genome)
```

Arguments

x	object of class data frame with columns 'chromosome', 'position', 'n_vaf', 't_vaf', 'sample'
y	object of class data frame with columns 'chromosome', 'start', 'end' specifying chromosomal boundaries for a genome assembly (required if genome is not specified)
genome	character string specifying the genome assembly from which input data is based

Value

list of inputs passing basic quality controls

lohSpec_slidingWindow *Obtain LOH data*

Description

Obtain LOH heatmap on entire chromsomes from samples in a cohort

Usage

```
lohSpec_slidingWindow(loh_data, step, window_size, normal)
```

Arguments

loh_data	data frame with columns "chromosome", "position", "n_vaf", "t_vaf", "sample" giving raw vaf calls for germline variants
step	integer with the length of divisions (bp) in chromosomes
window_size	integer with the size of the sliding window (bp) to be applied
normal	integer specifying the normal VAF frequency used in LOH calculations

Value

object of class dataframe containing LOH data

lohSpec_stepCalc *Obtain average loh within each step*

Description

Calculate avverage LOH within each step

Usage

```
lohSpec_stepCalc(final_dataset, step)
```

Arguments

final_dataset	object of class dataframe with columns 'window_start', 'window_stop', 'chromosome', 'position', 'n_vaf', 't_vaf', 'sample', and 'loh_diff_avg'
step	integer with the length of divisions (bp) in chromosomes

Value

list containing avg loh calculations for each step interval

lohSpec_tileCalc *Calculate loh difference*

Description

Obtain LOH on an entire chromsomes from samples in a cohort

Usage

```
lohSpec_tileCalc(window_data, normal)
```

Arguments

window_data	object of class data frame with columns "chromosome", "position", "n_vaf", "t_vaf", "sample", "bin", "window_start", "window_stop"
normal	integer specifying the subtraction value from tumor VAF

Value

object of class dataframe containing mean LOH difference calculations and column names "window_start", "window_stop", "chromosome", "position", "n_vaf", "t_vaf", "sample", "loh_diff"

lohSpec_tilePosition *Obtain window information*

Description

Calculate window positions to perform LOH calculation

Usage

```
lohSpec_tilePosition(out, window_size)
```

Arguments

out	object of class dataframe with columns 'chromosome', 'position', 'n_vaf', 't_vaf', and 'sample'
window_size	integer with the size of the sliding window (bp) to be applied

Value

list containing window start/stop positions for each chromosome from each sample to perform LOH calculations

lohSpec_tileWindow *Obtain LOH data*

Description

Obtain LOH heatmap on entire chromsomes from samples in a cohort

Usage

```
lohSpec_tileWindow(loh_data, window_size, normal)
```

Arguments

loh_data	data frame with columns "chromosome", "position", "n_vaf", "t_vaf", "sample" giving raw vaf calls for germline variants
window_size	integer with the size of the sliding window (bp) to be applied
normal	integer specifying the normal VAF frequency used in LOH calculations

Value

object of class dataframe containing LOH data

lohSpec_windowPosition *Obtain window information*

Description

Calculate window positions to perform LOH calculation

Usage

```
lohSpec_windowPosition(out, step, window_size)
```

Arguments

out	object of class dataframe with columns 'chromosome', 'position', 'n_vaf', 't_vaf', and 'sample'
step	integer with the length of divisions (bp) in chromosomes
window_size	integer with the size of the sliding window (bp) to be applied

Value

list containing window start/stop positions for each chromosome from each sample to perform LOH calculations

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 co-ordinate, 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)
```

---

lohView_buildMain *construct loh plot*

Description

given a loh data frame plot points in ggplot

Usage

```
lohView_buildMain(x, y, chr, layers = NULL)
```

Arguments

x	a data frame with columns chromosome, position, n_vaf, t_vaf, sample
y	a data frame with columns chromosome, coordinate for plotting chromosome boundaries
chr	a character string specifying chromosome
layers	additional ggplot2 layers to add

Value

ggplot2 object

---

lohView_qual *check input to lohView*

Description

Perform a data quality check for inputs to lohView

Usage

```
lohView_qual(x, y, genome)
```

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".
y	a data frame with columns "chrom", "chromStart", "chromEnd", "name", "gieStain"
genome	character string specifying UCSC genome to use

Value

a list of data frames passing quality checks

`lolliplot` *Construct a lolliplot*

Description

Given a data frame construct a plot displaying mutations on a transcript framework.

Usage

```
lolliplot(x, y = NULL, z = NULL, fillCol = NULL, labelCol = NULL,
  txtAngle = 45, txtSize = 5, pntSize = 4, proteinColour = "#999999",
  obsA.rep.fact = 5000, obsA.rep.dist.lmt = 500, obsA.attr.fact = 0.1,
  obsA.adj.max = 0.1, obsA.adj.lmt = 0.5, obsA.iter.max = 50000,
  obsB.rep.fact = 5000, obsB.rep.dist.lmt = 500, obsB.attr.fact = 0.1,
  obsB.adj.max = 0.1, obsB.adj.lmt = 0.5, obsB.iter.max = 50000,
  sideChain = FALSE, species = "hsapiens", maxLolliStack = NULL,
  plotLayer = NULL, paletteA = NULL, paletteB = NULL,
  host = "www.ensembl.org", out = "plot")
```

Arguments

`x`	Object of class data frame with rows representing mutations. The data frame must contain columns with the following names "transcript_name", "gene", and "amino_acid_change". Values in the "transcript_name" column must represent an ensembl transcript id and values in the "amino_acid_change" column must be in p.notation (see details).
`y`	Object of class data frame with rows representing mutations. The data frame must contain columns with the following names "transcript_name" and "amino_acid_change". Values in the "transcript_name" column must represent an ensembl transcript id and values in the "amino_acid_change" column must be in p. notation (optional, see details).
`z`	Object of class data frame with rows representing regions of interest. The data frame must contain columns with the following names "description", "start", "stop" (optional see details).
`fillCol`	Character string specifying the column name of the argument supplied to parameter x on which to colour the lollis representing mutations (see details).
`labelCol`	Character string specifying the column name of the argument supplied to parameter x from which to extract and display text corresponding to mutations (see details).
`txtAngle`	Integer specifying the angle of label text to be plotted if an argument is supplied to the labelCol parameter.
`txtSize`	Integer specifying the size of label text to be plotted if an argument is supplied to the labelCol parameter.
`pntSize`	Integer specifying the size of lolli points representing mutations.
`proteinColour`	Character string specifying the background colour of the protein.

obsA.rep.fact Numeric value representing the repulsive factor for the lollis plotted, which were derived from the argument supplied to parameter x (see details and vignette).

obsA.rep.dist.lmt Numberic value representing the repulsive distance limit for the lollis plotted, which were derived from the argument supplied to parameter x (see details and vignette).

obsA.attr.fact Numeric value representing the attraction factor for the lollis plotted, which were derived from the argument supplied to parameter x (see details and vignette).

obsA.adj.max Numeric value representing the max position adjustment for the lollis plotted, which were derived from the argument supplied to parameter x (see details and vignette).

obsA.adj.lmt Numeric value representing the adjustment limit for the lollis plotted, which were derived from the argument supplied to parameter x (see details and vignette).

obsA.iter.max Integer representing the number of iterations of position adjustments for the lollis plotted, which were derived from the argument supplied to parameter x (see details and vignette).

obsB.rep.fact Numeric value representing the repulsive factor for the lollis plotted, which were derived from the argument supplied to parameter y (see details and vignette).

obsB.rep.dist.lmt Numberic value representing the repulsive distance limit for the lollis plotted, which were derived from the argument supplied to parameter y (see details and vignette).

obsB.attr.fact Numeric value representing the attraction factor for the lollis plotted, which were derived from the argument supplied to parameter y (see details and vignette).

obsB.adj.max Numeric value representing the max position adjustment for the lollis plotted, which were derived from the argument supplied to parameter y (see details and vignette).

obsB.adj.lmt Numeric value representing the adjustment limit for the lollis plotted, which were derived from the argument supplied to parameter y (see details and vignette).

obsB.iter.max Integer representing the number of iterations of position adjustments for the lollis plotted, which were derived from the argument supplied to parameter y (see details and vignette).

sideChain Boolean specifying if amino acid sidechain data should be plotted in lieu of protein domains (see details).

species A valid species from which to retrieve protein domain and sequence data for a given transcript (see details).

maxLolliStack Integer specifying the cutoff for the maximum number of lollis allowed to be stacked at a single position.

plotLayer Valid ggplot2 layer to be added to the plot.

paletteA Character vector specifying colours for protein domains, valid only if sideChain==FALSE.

paletteB	Character vector specifying colours for lollis representing mutations, valid only if argument is supplied to fillCol.
host	Host to connect to for biomaRt queries (see details).
out	Character vector specifying the the object to output, one of "data", "grob", or "plot", defaults to "plot" (see returns).

Details

lolliplot is a function designed to display mutation information in the context of a protien identified by an ensembl transcript id. The lolliplot function will query ensembl via biomart to retrieve sequence and domain information in order to construct a representation of a protein and therefore requires an internet connection. A value must be supplied to the species parameter (defaults to hsapiens) in order for a successful biomart query. Valid arguments to this field are those species with datasts available via ensembl. please specify species in lowercase without a period (i.e. hsapiens instead of H.sapiens), lolliplot will inform the user of available species if input to the species parameter is not recognized. Further lolliplot will build a protein framework based on sequence data obtained from biomaRt, by default this will default to the latest ensembl version. In order for the most accurate representation the annotation version of the mutations given to lolliplot should match the annotation version used by biomaRt. The annotation version used by biomaRt can be changed via the host paramter (see vignette for more details).

lolliplot is capable of plotting two seperate sets of data on the protein representation specified by parameters ‘x‘ and ‘y‘, the data supplied to these parameters will be plotted on the top and bottom of the protein respectively. Note that input to these parameters is expected to correspond to a single ensembl transcript and that values in the "amino_acid_change" columns are required to be in p. notation (i.e. p.V600E). Further lolliplot is able to plot custom domain annotation if supplied via the parameter ‘z‘, this will override domain information obtained from biomart.

lolliplot uses a forcefield model from the package FField to attract and repulse lollis. The parameters for this force field model are set to reasonable defaults however may be adjusted via the obsA... and obsB... family of parameters. Please see the package FField available on cran for a description of these parameters. Note that the time to construct the lolliplot will in large part depend on the number of mutations and the values supplied to the forcefield parameters.

Value

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

Examples

```
# Create input data
data <- brcaMAF[brcaMAF$Hugo_Symbol == 'TP53',c('Hugo_Symbol', 'amino_acid_change_WU')]
data <- as.data.frame(cbind(data, 'ENST00000269305'))
colnames(data) <- c('gene', 'amino_acid_change', 'transcript_name')

# Call lolliplot
lolliplot(data)
```

lolliplot_AA2sidechain *Convert AA to side chain classification*

Description

Given the 1 letter code an amino acid, return the side chian classification

Usage

```
lolliplot_AA2sidechain(x)
```

Arguments

x	Character of length 1 giving the 1 letter amino acid code

Value

Object of class character

lolliplot_buildMain *Construct Lolliplot*

Description

Construct Lolliplot given gene and mutation data

Usage

```
lolliplot_buildMain(gene_data, length, mutation_observed, mutation_observed2,
  fill_value, label_column, plot_text_angle, plot_text_size, point_size,
  gene_colour, sequence_data, plot_sidechain = FALSE, layers = NULL,
  paletteA = NULL, paletteB = NULL)
```

Arguments

gene_data	object of class dataframe giving protien domain and gene information
length	integer specifying the length of the protien in amino acids
mutation_observed	object of class data frame specifying mutations observed in input file
mutation_observed2	optional object of class data frame specifying additional mutations for bottom track
fill_value	character string specifying the column on which to colour mutation points

label_column	character string specifying the column containing the labels to attach to mutation points
plot_text_angle	numeric value specifying the angle of text to be plotted
plot_text_size	numeric value specifying the size of text to be plotted
point_size	numeric value specigying the size of mutation points
gene_colour	color to shade plotted gene
sequence_data	object of class dataframe giving AA sequence, sidechain, and coord required if plot_sidechain is true
plot_sidechain	boolean specifying whether to plot the AA sidechain instead of domain information
layers	additional ggplot2 layers to plot
paletteA	Character vector specifying colours for gene features
paletteB	Character vector specifying colours for lolli features

Value

a ggplot2 object

`lolliplot_Codon2AA` *Convert Codon to AA*

Description

Convert a Codon to the appropriate amino acid

Usage

```
lolliplot_Codon2AA(x)
```

Arguments

x	Character string of length 1 giving the DNA codon to convert

Value

Character corresponding to the residue for the given codon

lolliplot_constructGene *Construct gene information*

Description

Build gene for input into lolliplot_buildMain

Usage

```
lolliplot_constructGene(gene, domain_data, length)
```

Arguments

gene	character string specifying gene name
domain_data	object of class data frame specifying protien domain information, obtained from lolliplot_fetchDomain, should contain columns giving "description", "start", "end"
length	integer specifying length of transcript in amino acids

Value

object of class data frame giving gene and domain information

lolliplot_DNAconv *Convert DNA character string*

Description

Convert a character string of nucleotides to amino acids or side chain class

Usage

```
lolliplot_DNAconv(x, to = "residue")
```

Arguments

x	Character string of nucleotides to convert
to	Character string specifying conversion to do, one of "codon", "residue", "sidechain"

Value

Converted string of nucleotides as character vector

lolliplot_dodgeCoordX *dodge coordinates*

Description

given amino acid position dodge on x axis

Usage

```
lolliplot_dodgeCoordX(x, rep.fact = 5000, rep.dist.lmt = 500,
  attr.fact = 0.1, adj.max = 0.1, adj.lmt = 0.5, iter.max = 50000)
```

Arguments

x	numeric vector of position coordinates on x axis
rep.fact	repulsive factor for plotted mutations observed track
rep.dist.lmt	repulsive distance limit for plotted mutations observed track
attr.fact	attraction factor for plotted mutations observed track
adj.max	maximum position change for each iteration observed track
adj.lmt	position adjustment limit which simulation stops observed track
iter.max	maximum iterations beyond which to stop the simulation observed track

Value

numeric vector of dodged position coordinates on x axis

lolliplot_dodgeCoordY *dodge coordinates*

Description

given a data frame, dodge x coordinates ontop of each other

Usage

```
lolliplot_dodgeCoordY(x, track = "top")
```

Arguments

x	data frame containing columns coord_x_dodge
track	character vector, one of "top", "bottom" specifying whether to dodge in a positive or negative fashion

Value

numeric vector of dodged position coordinates on y axis

lolliplot_fetchDomain *fetch protein domains*

Description

Retrieve protein domains given ensembl transcript ID

Usage

```
lolliplot_fetchDomain(transcriptID, species = "hsapiens",
  host = "www.ensembl.org")
```

Arguments

transcriptID	String specifying ensembl transcript id
species	character string to use when searching for ensemblMart dataset
host	Host to connect to.

Value

data frame of protien domains and start/stop coordinates

lolliplot_mutationObs *format mutation observations*

Description

Create a data frame of mutation observations

Usage

```
lolliplot_mutationObs(x, track, fill_value, label_column, rep.fact,
  rep.dist.lmt, attr.fact, adj.max, adj.lmt, iter.max)
```

Arguments

x	object of class data frame with columns trv_type and amino acid change
track	character string specifying one to 'top', 'bottom' to specify proper track
fill_value	character string giving the name of the column to shade variants on
label_column	character string specifying column containing text information to be plotted
rep.fact	repulsive factor for plotted mutations observed track
rep.dist.lmt	repulsive distance limit for plotted mutations observed track
attr.fact	attraction factor for plotted mutations observed track
adj.max	maximum position change for each iteration observed track
adj.lmt	position adjustment limit which simulation stops observed track
iter.max	maximum iterations beyond which to stop the simulation observed track

Value

object of class data frame giving mutation observations

---

lolliplot_qual *Check input to lolliplot*

Description

Perform Basic quality checks for lolliplot input

Usage

```
lolliplot_qual(x, y, z)
```

Arguments

x	object of class data frame containing columns transcript_name, gene, and amino_acid_change and rows denoting mutations
y	object of class data frame containing columns transcript_name, and amino_acid_change and rows denoting mutations
z	Object of class data frame containing columns "description", "start", "stop" specifying gene regions to highlight

Value

objects passing basic quality checks

---

lolliplot_reduceLolli *Reduce Lolli*

Description

Reduce lollis stacked ontop of each other to the amount specified

Usage

```
lolliplot_reduceLolli(x, max = NULL)
```

Arguments

x	Data frame with column name mutation_coord to reduce lollis on
max	Integer specifying the maximum number of lollis to allow

Value

Object of class data frame taking the reduced form of x

lolliplot_transcriptID2codingSeq *fetch protein length*

Description

Retrieve protein length from ensembl database given enseml transcript id

Usage

```
lolliplot_transcriptID2codingSeq(transcriptID, species = "hsapiens",
  host = "www.ensembl.org")
```

Arguments

transcriptID	character string giving ensembl transcript id
species	character string to use when searching for ensemblMart dataset
host	Host to connect to.

Value

length in residues of ensembl transcript id

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

multi_align *align CN/LOH plots on x axis*

Description

given a chromosome and CN/LOH plot align plot widths

Usage

```
multi_align(p1, p2)
```

Arguments

p1	ggplot object of chromosome
p2	ggplot object of CN or LOH

Value

ggplot object

multi_buildClin *plot clinical information*

Description

given a data frame with columns names sample, variable, and value create a ggplot2 object

Usage

```
multi_buildClin(x, clin.legend.col = 1, clin.var.colour = NULL,
  clin.var.order = NULL, clin.layers = NULL)
```

Arguments

x	a data frame in "long" format giving additional information to be plotted, requires columns "sample", "variable", and "value"
clin.legend.col	an integer specifying the number of columns to plot in the legend
clin.var.colour	a named character vector specifying the mapping between colors and variables
clin.var.order	a character vector of variables to order the legend by
clin.layers	additional ggplot2 layers to plot

Value

a grob object

multi_chrBound *retrieve and format CN_cohort plot supplemental data*

Description

given a genome obtain Start and Stop positions for all chromosomes in the genome

Usage

```
multi_chrBound(x)
```

Arguments

x	data frame containg columns chromosome, start, end

Value

object of class data frame formatted to internal specifications

multi_cytobandRet *Retrieve cytogenetic bands*

Description

given a genome query UCSC for cytogenetic band locations

Usage

```
multi_cytobandRet(genome)
```

Arguments

genome	character string giving a UCSC genome

Value

object of class data frame

multi_selectOut *Choose output*

Description

Selector for choosing output for GenVisR functions

Usage

```
multi_selectOut(data, plot, out = "plot", draw = "FALSE")
```

Arguments

data	Data object to output
plot	Plot object to output
out	Character vector specifying the the object to output, one of "data", "grob", or "plot".
draw	Boolean specifying if the input to plot needs to be drawn

Value

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

multi_subsetChr *subset based on chr*

Description

given a data frame subset out specific a chromosome

Usage

```
multi_subsetChr(x, chr)
```

Arguments

x	a data frame with columns chromosome
chr	character string specifying UCSC chromosome to subset on

Value

object of class data frame

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")
```

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 givin 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 transverions 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).

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")
```

TvTi_alignPlot *align TvTi plots on y axis*

Description

align transition/transversion plots

Usage

```
TvTi_alignPlot(p1 = NULL, p2 = NULL, p3 = NULL)
```

Arguments

p1	main plot
p2	left expected value subplot
p3	bottom clinical subplot

Value

ggplot object

TvTi_annoTransTranv *Annotate Transitions and Transversions*

Description

Given a data frame with columns reference and variant annotate the base change occurring

Usage

```
TvTi_annoTransTranv(x)
```

Arguments

x	Object of class data frame containing columns 'reference', 'variant'

Value

Object of class data frame with transition/transversion annotations appended

TvTi_buildMain *build transitions/transversions*

Description

Given a data frame with columns 'trans_tranv', 'sample', 'Freq', and 'Prop', build a transition/transversion plot

Usage

```
TvTi_buildMain(x, y = NULL, type = "Proportion", label_x_axis = TRUE,
  x_axis_text_angle = 45, palette = c("#D53E4F", "#FC8D59", "#FEE08B",
  "#E6F598", "#99D594", "#3288BD"), plot_expected = FALSE,
  tvti.layers = NULL, expec.layers = NULL, title_x_axis = TRUE)
```

Arguments

x	Object of class data frame containing columns 'trans_tranv', 'sample', 'Freq', and 'Prop'
y	Object of class data frame containing columns 'Prop', 'trans_tranv' for display of expected results
type	Object of class character specifying whether to plot the Proportion or Frequency, one of "Prop"
label_x_axis	boolean specifying wheter to label x axis
x_axis_text_angle	Integer specifying the angle to labels on x_axis
palette	Character vector of length 6 specifying colors for trans/tranv type
plot_expected	Boolean specifying if this is the main TvTi plot or a sub plot for expected values
tvti.layers	Additional ggplot2 layers for the main plot
expec.layers	Additional ggplot2 layers for the expected values plot
title_x_axis	boolean specifying whether to display an x axis title

Value

GGplot Object

TvTi_calcTransTranvFreq *Calculate Transition/Transversion Frequency*

Description

Given a data frame with columns reference, variant, sample, and trans/tranv calculate the frequencies of transitions and transversion occuring.

Usage

```
TvTi_calcTransTranvFreq(x)
```

Arguments

x	Object of class data frame containing columns 'reference', 'variant', 'sample', 'trans_tranv'

Value

Object of class data frame with Frequency and Proportion of Transistions/Transversions appended on a sample level

TvTi_convMAF *Convert .maf format to internal format*

Description

Convert data frame in .maf format to an internally recogized format

Usage

```
TvTi_convMaf(x)
```

Arguments

x	Object of class data frame containing columns 'Tumor_Sample_Barcode', 'Reference_Allele' 'Tumor_Seq_Allele1', 'Tumor_Seq_Allele2'

Value

a data frame, with column names 'sample', 'reference', 'variant'

TvTi_qual *Check input to TvTi*

Description

Perform quality check for input to function TvTi

Usage

```
TvTi_qual(x, y = NULL, z = NULL, file_type = "MAF")
```

Arguments

x	Object of class data frame containing columns 'sample', reference', 'variant' for 'MGI' file or 'Tumor_Sample_Barcode', 'Reference_Allele', 'Tumor_Seq_Allele1', 'Tumor_Seq_Allele2' for 'MAF' file
y	Object of class data frame containing columns "Prop", "trans_tranv"
z	Object of class data frame containing columns "sample", "variable", "value" denoting clinical information
file_type	Character string spedifying th input file type expected

Value

a data frame, or list of data frames passing quality checks

TvTi_rmIndel *Remove indels*

Description

Given a data frame with columns reference and variants remove all indels from data

Usage

```
TvTi_rmIndel(x)
```

Arguments

x	Object of class data frame containing columns 'reference', 'variant'

Value

Object of class data frame with indels removed

TvTi_rmMnuc *Remove multinucleotide codes*

Description

Given a data frame with columns reference and variants remove all multinucleotides from data

Usage

```
TvTi_rmMnuc(x)
```

Arguments

x	Object of class data frame containing columns 'reference', 'variant'

Value

Object of class data frame with multi nucleotide codes removed

waterfall *Construct a waterfall plot*

Description

Given a data frame construct a water fall plot showing the mutation burden and mutation type on a gene and sample level.

Usage

```
waterfall(x, mainRecurCutoff = 0, mainGrid = TRUE, mainXlabel = FALSE,
  main_geneLabSize = 8, mainLabelCol = NULL, mainLabelSize = 4,
  mainLabelAngle = 0, mainDropMut = FALSE, mainPalette = NULL,
  mainLayer = NULL, mutBurden = NULL, plotMutBurden = TRUE,
  coverageSpace = 44100000, mutBurdenLayer = NULL, clinData = NULL,
  clinLegCol = 1, clinVarOrder = NULL, clinVarCol = NULL,
  clinLayer = NULL, sampRecurLayer = NULL, plotGenes = NULL,
  geneOrder = NULL, plotSamples = NULL, sampOrder = NULL,
  maxGenes = NULL, rmvSilent = FALSE, fileType = "MAF",
  variant_class_order = NULL, out = "plot")
```

Arguments

x	Object of class data frame representing annotated mutations. The data frame supplied must have one of the following sets of column names ("Tumor_Sample_Barcode", "Hugo_Symbol", "Variant_Classification") for fileType="MAF", ("sample","gene_name","trv_type") for fileType="MGI" or ("sample", "gene", "variant_class") for fileType="Custom". This columns should represent samples in a cohort, gene with mutation, and the mutation type respectively.
mainRecurCutoff	Numeric value between 0 and 1 specifying a mutation recurrence cutoff. Genes which do not have mutations in the proportion os samples defined are removed.
mainGrid	Boolean specifying if a grid should be overlayed on the main plot. Not recommended if the number of genes or samples to be plotted is large.
mainXlabel	Boolean specifying whether to label the x-axis with sample names. Not recommended if the number of samples to be plotted is large.
main_geneLabSize	Intenger specifying the size of gene names displayed on the y-axis.
mainLabelCol	Character string specifying a column name from the argument supplied to parameter ‘x‘ from which to derive cell labels from (see details and vignette).
mainLabelSize	Integer specifying the size of text labels for cells in the main plot. Valid only if argument is supplied to the parameter ‘mainLabelCol‘.
mainLabelAngle	Integer specifying the degree of rotation for text labels. Valid only if argument is supplied to the parameter ‘mainLabelCol‘.
mainDropMut	Boolean specifying whether to drop unused "mutation type" levels from the legend.
mainPalette	Character vector specifying colours for mutation types plotted in the main plot, must specify a colour for each mutation type plotted.
mainLayer	Valid ggplot2 layer to be added to the main plot.
mutBurden	Object of class data frame containing columns "sample", "mut_burden" with sample levels matching those supplied in x.
plotMutBurden	Boolean specify if the mutation burden sub-plot should be displayed.
coverageSpace	Integer specifying the size in bp of the genome covered by sequence data from which mutations could be called (see details and vignette).
mutBurdenLayer	Valid ggplot2 layer to be added to the top sub-plot.
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).
clinLegCol	Integer specifying the number of columns in the legend for the clinical data, only valid if argument is supplied to parameter clinData.
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).

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").
clinLayer	Valid ggplot2 layer to be added to the clinical sub-plot.
sampRecurLayer	Valid ggplot2 layer to be added to the left sub-plot.
plotGenes	Character vector specifying genes to plot. If not null genes not specified within this character vector are removed.
geneOrder	Character vector specifying the order in which to plot genes.
plotSamples	Character vector specifying samples to plot. If not null all other samples not specified within this parameter are removed.
sampOrder	Character vector specifying the order of the samples to plot.
maxGenes	Integer specifying the maximum number of genes to be plotted. Genes kept will be choosen based on the reccurence of mutations in samples.
rmvSilent	Boolean specifying if silent mutations should be removed from the plot.
fileType	Character string specifying the file format of the data frame specified to parameter ‘x‘, one of "MGI", "MAF", "Custom" (see details and vignette).
variant_class_order	Character vector specifying the hierarchical order of mutation types to plot, required if file_type == "Custom" (see details and vignette).
out	Character vector specifying the the object to output, one of "data", "grob", or "plot", defaults to "plot" (see returns).

Details

waterfall is a function designed to visualize the mutations seen in a cohort. The function takes a data frame with appropriate column names (see fileType parameter) and plots the mutations within. In cases where multiple mutations occur in the same cell the most deleterious mutation is given priority (see vignette for default priority). If the fileType parameter is set to "Custom" the user most supply this priority via the ‘variant_class_order‘ parameter with the highest priorities occuring first. Additionally this parameter will override the default orders of MGI and MAF file types.

Various data subsets are allowed via the waterfall function (see above), all of these subsets will occur independently of the mutation burden calculation. To clarify the removal of genes and mutations will only occur after the mutation burden is calculated. The mutation burden calculation is only meant to provide a rough estimate and assumes that the coverage breadth within the cohort is aproximately equal. For more accurate calculations it is recommended to supply this information via the mutBurden parameter which. Note that the mutation burden calculation relies on the ‘coverageSpace‘ parameter (see vignette).

It is possible to display additional information within the plot via cell labels. The ‘mainLabelCol‘ parameter will look for an additional column in the data frame and plot text within cells based on those values (see vignette).

Value

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

Examples

```
# Plot the data
waterfall(brcaMAF, plotGenes=c("PIK3CA", "TP53", "USH2A", "MLL3", "BRCA1"))
```

`waterfall_align` *align plots*

Description

align mutation landscape, mutation burden on sample, and mutation burden on gene plots

Usage

```
waterfall_align(p2, p1, p3, p4)
```

Arguments

`p2`	ggplot object displaying mutation burden on gene
`p1`	ggplot object displaying a mutation landscape
`p3`	ggplot object displaying mutation burden on sample
`p4`	ggplot object displaying clinical information "optional"

Value

a grob object

`waterfall_buildGenePrevelance`
plot mutation recurrence in genes

Description

plot a bar graph displaying the percentage of samples with a mutation

Usage

```
waterfall_buildGenePrevelance(data_frame, layers = NULL)
```

Arguments

`data_frame`	a data frame in MAF format
`layers`	additional ggplot2 layers

Value

a ggplot object

`waterfall_buildMain` *Plot a mutation heatmap*

Description

Plot a Mutation Landscape with variables sample, gene, mutation

Usage

```
waterfall_buildMain(data_frame, grid = TRUE, label_x = FALSE,
  gene_label_size = 8, file_type = "MGI", drop_mutation = FALSE,
  plot_x_title = TRUE, plot_label = FALSE, plot_label_size = 4,
  plot_palette = NULL, layers = NULL, plot_label_angle = 0)
```

Arguments

`data_frame`	a data frame in MAF format
`grid`	boolean value whether to overlay a grid on the plot
`label_x`	boolean value whether to label the x axis
`gene_label_size`	numeric value indicating the size of the gene labels on the y-axis
`file_type`	character string specifying the file type, one of 'MAF' or 'MGI'
`drop_mutation`	Boolean specifying whether to drop unused "mutation type" levels from the legend
`plot_x_title`	Boolean specifying whether to plot the x_axis title
`plot_label`	Boolean specifying whether to plot text inside each cell
`plot_label_size`	Integer specifying text size of cell labels
`plot_palette`	Character vector specifying colors to fill on mutation type
`layers`	additional ggplot2 layers to plot
`plot_label_angle`	angle at which to plot label text if plot_label is true

Value

a ggplot2 object

waterfall_buildMutBurden_A

plot mutation burden

Description

plot a barchart showing mutations per MB

Usage

```
waterfall_buildMutBurden_A(x, coverage_space, layers = NULL)
```

Arguments

x	a data frame in MAF format
coverage_space	an integer specifying the coverage space in base pairs from which a mutation could occur
layers	Additional ggplot2 layers to plot

Value

a ggplot object

waterfall_buildMutBurden_B

plot mutation burden

Description

plot a barchart showing mutation burden given by data frame

Usage

```
waterfall_buildMutBurden_B(x, layers = NULL)
```

Arguments

x	a data frame containing columns sample, mut_burden
layers	additional ggplot2 layers to plot

Value

a ggplot object

waterfall_calcMutFreq *Calculate Synonymous/Nonsynonymous mutation frequency*

Description

Creates a data frame giving synonymous/nonsynonymous counts on a sample level

Usage

```
waterfall_calcMutFreq(x)
```

Arguments

x	data frame in long format with columns sample, trv_type

Value

a data frame with synonymous/nonsynonymous counts appended

waterfall_Custom2anno *Convert Custom File*

Description

Convert columns of a Custom annotation file into a format recognizable by internal functions

Usage

```
waterfall_Custom2anno(x, label_col)
```

Arguments

x	a data frame with columns having values for sample, gene, mutation type
label_col	Character string specifying the column name of a label column (optional)

Value

a data frame coerced from custom to annotation format

waterfall_geneAlt *mutation sample cutoff gene based*

Description

Subset a internal mutSpec file keeping only samples within the specified gene list

Usage

```
waterfall_geneAlt(x, genes)
```

Arguments

x	a data frame in long format with columns 'gene', 'trv_type'
genes	character vector listing genes to plot

Value

a subset data frame

waterfall_geneRecurCutoff *Mutation Recurrence Cutoff*

Description

Subset a MAF file keeping only samples that meet a mutation recurrence cutoff

Usage

```
waterfall_geneRecurCutoff(x, recurrence_cutoff)
```

Arguments

x	data frame in long format with columns 'gene', 'trv_type', 'sample'
recurrence_cutoff	integer specifying removal of entries not seen in at least "x" percent of samples

Value

a subset data frame

waterfall_geneSort *sort waterfall file by gene*

Description

order a waterfall file ranking genes with more mutations higher if a gene order is unspecified.

Usage

```
waterfall_geneSort(x, geneOrder = NULL)
```

Arguments

x	Data frame with columns names "gene", "trv_type".
geneOrder	Character vector specifying the order in which to plot genes.

Value

Character vector of ordered genes

waterfall_hierarchyTRV *Hiearchical removal of MAF entries*

Description

Remove MAF entries with the same gene/sample in an ordered fashion such that the most deleterious are retained

Usage

```
waterfall_hierarchyTRV(x, file_type, variant_class_order)
```

Arguments

x	a data frame in long format with columns sample, gene, trv_type
file_type	The type of file to act on one of 'MAF", "MGI", "Custom"
variant_class_order	character vector giving the hierarchical order of mutation types to plot

Value

a data frame with multiple mutations in the same sample/gene collapsed on the most deleterious

waterfall_MAF2anno *Convert MAF File*

Description

Convert columns of a mutation annotation file "MAF" into a format recognizable by internal functions

Usage

```
waterfall_MAF2anno(x, label_col)
```

Arguments

x	a data frame in MAF format
label_col	Character string specifying the column name of a label column

Value

a data frame coerced from MAF to TGI format

waterfall_MGI2anno *Convert MGI File*

Description

Convert columns of a mutation annotation file "MGI" into a format recognizable by internal functions

Usage

```
waterfall_MGI2anno(x, label_col)
```

Arguments

x	a data frame in MGI internal format
label_col	Character string specifying the column name of a label column

Value

a data frame coerced from MGI to internal annotation format

waterfall_NA2gene *Assign NA samples a gene*

Description

Replace NA values in a gene column with the top gene name

Usage

```
waterfall_NA2gene(x)
```

Arguments

x	a data frame in anno format

Value

a data frame with NA values in a gene column coerced to the top gene name

waterfall_qual *Check input to mutSpec*

Description

Perform a data quality check on input to mutSpec

Usage

```
waterfall_qual(x, y, z, file_type, label_col)
```

Arguments

x	a data frame in annotation format
y	a data frame containing clinical data or a null object
z	a data frame containing mutation burden information or a null object
file_type	Character string specifying the input format to expect in x
label_col	Character string specifying the column name of a label column

Value

a list of data frames passing quality checks

waterfall_rmvSilent *Silent Mutation Removal*

Description

Subset a MAF file setting keeping only sample information if a mutation is silent

Usage

```
waterfall_rmvSilent(x)
```

Arguments

x	a data frame with columns 'sample', 'gene', 'trv_type'

Value

a subset data frame

waterfall_sampAlt *mutation sample subset sample based*

Description

Alter a mutSpec input file keeping/adding entries in a selection of samples

Usage

```
waterfall_sampAlt(x, samples)
```

Arguments

x	a data frame in long format with columns 'sample', 'trv_type'
samples	character vector giving samples to plot

Value

a subset data frame

`waterfall_sampSort` *sort samples in an internal waterfall file.*

Description

perform a hiearchial sort on samples based on the presence of mutations in an ordered list of genes if a sample order is unspecified.

Usage

```
waterfall_sampSort(x, sampOrder = NULL)
```

Arguments

`x`	a data frame in long format with column names "sample", "gene", "trv_type"
`sampOrder`	Character vector specifying the order of samples to plot.

Value

a vector of samples in a sorted order

Index