Package ‘qtl2ggplot’

May 26, 2021

Version 1.0.6
Date 2021-05-25
Title Data Visualization for QTL Experiments
Description Functions to plot QTL (quantitative trait loci) analysis results and related diagnostics.
  Part of 'qtl2'; an upgrade of the 'qtl' package to better handle high-dimensional data and complex cross designs.
Depends R (>= 3.1.0)
Imports Rcpp (>= 0.12.7), assertthat, dplyr, ggplot2, purrr, stringr, tidyr, rlang, graphics, RColorBrewer, grid, qtl2
Suggests devtools, testthat, roxygen2, knitr, rmarkdown
VignetteBuilder knitr
License GPL-3
URL https://kbroman.org/qtl2/
LinkingTo Rcpp
RoxygenNote 7.1.1
NeedsCompilation yes
Author Brian S Yandell [aut, cre],
  Karl W Broman [aut]
Maintainer Brian S Yandell <brian.yandell@wisc.edu>
Repository CRAN
Date/Publication 2021-05-26 12:30:02 UTC

R topics documented:

  color_patterns_get ........................................ 2
  color_patterns_pheno ....................................... 2
  color_patterns_set ....................................... 3
  ggplot_coef ............................................... 4
color_patterns_pheno

Set up col, pattern, shape and group for plotting.

Description
Set up col, pattern, shape and group for plotting.

Usage

```r
color_patterns_pheno(
  scan1ggdata, lod, pattern, col, shape, patterns, facet = NULL
)
```

Arguments

- `scan1ggdata`: data frame to be used for plotting
- `lod`: Color for color column in scan1ggdata
- `pattern`: for colors (default NULL uses "Dark2" from RColorBrewer package)
- `col`: for colors (default NULL uses "Dark2" from RColorBrewer package)
- `palette`: for colors (default NULL uses "Dark2" from RColorBrewer package)
- `shape`: for colors (default NULL uses "Dark2" from RColorBrewer package)
- `patterns`: for colors (default NULL uses "Dark2" from RColorBrewer package)
- `facet`: for colors (default NULL uses "Dark2" from RColorBrewer package)

Value

list of colors and shapes.

color_patterns_get Set up col, pattern and group for plotting.

Description
Set up col, pattern and group for plotting.

Usage

```r
color_patterns_get(scan1ggdata, col, palette = NULL)
```

Arguments

- `scan1ggdata`: data frame to be used for plotting
- `col`: Color for color column in scan1ggdata
- `palette`: for colors (default NULL uses "Dark2" from RColorBrewer package)
**color_patterns_set**

**Arguments**

- `scan1ggdata`: data frame to be used for plotting
- `lod`: matrix of LOD scores by position and pheno
- `pattern`: allele pattern of form AB:CDEFGH
- `col`: Color for color column in `scan1ggdata`
- `shape`: Shape for shape column in `scan1ggdata`
- `patterns`: Connect SDP patterns: one of `c(“none”, “all”, “hilit”)`
- `facet`: use `facet_wrap` if not NULL

**Value**

data frame `scan1ggdata` with additional objects.

**Description**

Set up colors for patterns or points

**Usage**

```R
color_patterns_set(
  scan1output,
  snpinfo,
  patterns,
  col,
  pattern,
  show_all_snps,
  col_hilit,
  drop_hilit,
  maxlod
)
```

**Arguments**

- `scan1output`: output of linear mixed model for phename (see `scan1`)
- `snpinfo`: Data frame with snp information
- `patterns`: Connect SDP patterns: one of `c(“none”, “all”, “hilit”)`
- `col`: Color of other points, or colors for patterns
- `pattern`: allele pattern as of form AB:CDEFGH
- `show_all_snps`: show all SNPs if TRUE
- `col_hilit`: Color of highlighted points
- `drop_hilit`: SNPs with LOD score within this amount of the maximum SNP association will be highlighted.
- `maxlod`: Maximum LOD for drop of `drop_hilit`
ggplot_coef

Plot QTL effects along chromosome

Value

list of col and pattern.

Description

Plot estimated QTL effects along a chromosomes.

Usage

ggplot_coef(
  x,
  map,
  columns = NULL,
  col = NULL,
  scan1_output = NULL,
  gap = 25,
  ylim = NULL,
  bgcolor = "gray90",
  altbgcolor = "gray85",
  ylab = "QTL effects",
  xlim = NULL,
  ...
)

## S3 method for class 'scan1coef'
autoplot(x, ...)

Arguments

- **x**: Estimated QTL effects ("coefficients") as obtained from `scan1coef`.
- **map**: A list of vectors of marker positions, as produced by `insert_pseudomarkers`.
- **columns**: Vector of columns to plot.
- **col**: Vector of colors, same length as columns. If NULL, some default choices are made.
- **scan1_output**: If provided, we make a two-panel plot with coefficients on top and LOD scores below. Should have just one LOD score column; if multiple, only the first is used.
- **gap**: Gap between chromosomes.
- **ylim**: y-axis limits. If NULL, we use the range of the plotted coefficients.
ggplot_coef

bgcolor  Background color for the plot.
altbgcolor  Background color for alternate chromosomes.
ylab  y-axis label
xlim  x-axis limits. If NULL, we use the range of the plotted coefficients.
...  Additional graphics parameters.
colors  Colors to use for plotting.

Details

ggplot_coefCC() is the same as ggplot_coef(), but forcing columns=1:8 and using the Collaborative Cross colors, CCcolors.

Value

object of class ggplot.

See Also

ggplot_scan1, ggplot_snpasso

Examples

# load qtl2 package for data and genoprob calculation
library(qtl2)

# read data
iron <- read_cross2(system.file("extdata", "iron.zip", package="qtl2"))

# insert pseudomarkers into map
map <- insert_pseudomarkers(iron$gmap, step=1)

# calculate genotype probabilities
probs <- calc_genoprob(iron, map, error_prob=0.002)

# grab phenotypes and covariates; ensure that covariates have names attribute
pheno <- iron$pheno[,1]
covar <- match(iron$covar$sex, c("f", "m")) # make numeric
names(covar) <- rownames(iron$covar)

# calculate coefficients for chromosome 7
coef <- scan1coef(probs[,7], pheno, addcovar=covar)

# plot QTL effects
library(ggplot2)
autoplot(coef, map[7], columns=1:3, col=c("slateblue", "violetred", "green3"))
**Description**

Plot gene locations for a genomic interval, as rectangles with gene symbol (and arrow indicating strand/direction) below.

**Usage**

```r
ggplot_genes(
  genes,
  xlim = NULL,
  minrow = 4,
  padding = 0.2,
  colors = c("black", "red3", "green4", "blue3", "orange"),
  ...
)
```

## S3 method for class 'genes'

```r
autoplot(x, ...)
```

**Arguments**

- `genes`: Data frame containing start and stop in bp, strand (as "-", "+", or NA), and Name.
- `xlim`: x-axis limits (in Mbp)
- `minrow`: Minimum number of rows of genes
- `padding`: Proportion to pad with white space around the genes
- `colors`: Vectors of colors, used sequentially and then re-used.
- `...`: Optional arguments passed to `plot`
- `x`: Object of class `genes`

**Value**

None.

**Examples**

```r
genes <- data.frame(chr = c("6", "6", "6", "6", "6", "6", "6", "6"),
  start = c(139.988753, 140.680185, 141.708118, 142.234227, 142.587862,
    143.232344, 144.398099, 144.993835),
  stop = c(140.041457, 140.826797, 141.773810, 142.322981, 142.702315,
    143.260627, 144.399821, 145.076184),
  strand = c("-", "+", "-", "-", "-", NA, "+", "-"),
  Name = c("Plcz1", "Gm30215", "Gm5724", "Slco1a5", "Abcc9", "..."))
```
Description

Plot genes at positions

Usage

ggplot_genes_internal(
  start, end, strand,
  rect_top, rect_bottom, colors, space,
  y, dir_symbol, name,
  xlim, xlab = "Position (Mbp)",
  ylab = "",
  bgcolor = "gray92",
  xat = NULL,
  legend.position = "none",
  vlines = NULL,
  ...
)

Arguments

start, end, strand, rect_top, rect_bottom, colors, space, y, dir_symbol, name, xlim
  usual parameters

legend.position, vlines, xlab, ylab, bgcolor, xat
  hidden parameters

... Additional graphics parameters.

Value

object of class ggplot.
ggplot_onegeno

Plot one individual’s genome-wide genotypes

**Description**

Plot one individual’s genome-wide genotypes

**Usage**

```r
ggplot_onegeno(
  geno,  # Imputed phase-known genotypes, as a list of matrices (as produced by `maxmarg`) or a list of three-dimensional arrays (as produced by `guess_phase`).
  map,   # Marker map (a list of vectors of marker positions).
  ind = 1,  # Individual to plot, either a numeric index or an ID.
  chr = NULL,  # Selected chromosomes to plot; a vector of character strings.
  col = NULL,  # Vector of colors for the different genotypes.
  shift = FALSE,  # If TRUE, shift the chromosomes so they all start at 0.
  chrwidth = 0.5,  # Total width of rectangles for each chromosome, as a fraction of the distance between them.
  ...  # Additional graphics parameters
)
```

**Arguments**

- `geno`: Imputed phase-known genotypes, as a list of matrices (as produced by `maxmarg`) or a list of three-dimensional arrays (as produced by `guess_phase`).
- `map`: Marker map (a list of vectors of marker positions).
- `ind`: Individual to plot, either a numeric index or an ID.
- `chr`: Selected chromosomes to plot; a vector of character strings.
- `col`: Vector of colors for the different genotypes.
- `shift`: If TRUE, shift the chromosomes so they all start at 0.
- `chrwidth`: Total width of rectangles for each chromosome, as a fraction of the distance between them.
- `...`: Additional graphics parameters

**Value**

Object of class `ggplot`.

**Examples**

```r
# load qtl2 package for data and genoprob calculation
library(qtl2)

# read data
iron <- read_cross2(system.file("extdata", "iron.zip", package="qtl2"))

# insert pseudomarkers into map
```
map <- insert_pseudomarkers(iron$gmap, step=1)

# calculate genotype probabilities
probs <- calc_genoprob(iron, map, error_prob=0.002)

# inferred genotype at a 28.6 cM on chr 16
geno <- maxmarg(probs)

ggplot_onegeno(geno, map, shift = TRUE)
ggplot_onegeno(geno, map, ind=1:4)

---

**ggplot_peaks**

*Plot QTL peak locations*

**Description**

Plot QTL peak locations (possibly with intervals) for multiple traits.

**Usage**

```r
ggplot_peaks(
  peaks,
  map,
  chr = NULL,
  tick_height = 0.3,
  gap = 25,
  bgcolor = "gray90",
  altbgcolor = "gray85",
  ...
)
```

**Arguments**

- **peaks**: Data frame such as that produced by `find_peaks` containing columns `chr`, `pos`, `lodindex`, and `lodcolumn`. May also contain columns `ci_lo` and `ci_hi`, in which case intervals will be plotted.
- **map**: Marker map, used to get chromosome lengths (and start and end positions).
- **chr**: Selected chromosomes to plot; a vector of character strings.
- **tick_height**: Height of tick marks at the peaks (a number between 0 and 1).
- **gap**: Gap between chromosomes.
- **bgcolor**: Background color for the plot.
- **altbgcolor**: Background color for alternate chromosomes.
- **...**: Additional graphics parameters
Plot phenotype vs genotype for a single putative QTL and a single phenotype.
Usage

```r
ggplot_pxg(
genom, phenom,
sort = TRUE,
SEmult = NULL,
pooledSD = TRUE,
jitter = 0.2,
bgcolor = "gray90",
seg_width = 0.4,
seg_lwd = 2,
seg_col = "black",
hlines = NULL,
hlines_col = "white",
hlines_lty = 1,
hlines_lwd = 1,
vlines_col = "gray80",
vlines_lty = 1,
vlines_lwd = 3,
force_labels = TRUE,
alternate_labels = FALSE,
omit_points = FALSE,
...)
)

mean_pxg(genom, phenom, dataframe = NULL)
```

Arguments

genom Vector of genotypes, as produced by `maxmarg` with specific chr and pos.
phenom Vector of phenotypes.
sort If TRUE, sort genotypes from largest to smallest.
SEmult If specified, interval estimates of the within-group averages will be displayed, as mean +/- SE * SEmult.
pooledSD If TRUE and SEmult is specified, calculated a pooled within-group SD. Otherwise, get separate estimates of the within-group SD for each group.
jitter Amount to jitter the points horizontally, if a vector of length > 0, it is taken to be the actual jitter amounts (with values between -0.5 and 0.5).
bgcolor Background color for the plot.
seg_width Width of segments at the estimated within-group averages
seg_lwd Line width used to plot estimated within-group averages
seg_col Line color used to plot estimated within-group averages
hlines Locations of horizontal grid lines.
hlines_col Color of horizontal grid lines
hlines_lty Line type of horizontal grid lines
hlines_lwd  Line width of horizontal grid lines
vlines_col  Color of vertical grid lines
vlines_lty  Line type of vertical grid lines
vlines_lwd  Line width of vertical grid lines
force_labels  If TRUE, force all genotype labels to be shown.
alternate_labels  If TRUE, place genotype labels in two rows
omit_points  If TRUE, omit the points, just plotting the averages (and, potentially, the +/- SE intervals).
...  Additional graphics parameters, passed to plot.
dataframe  Supplied data frame, or constructed from geno and pheno if NULL.

Value
object of class ggplot.

See Also
plot_coef

Examples

# load qtl2 package for data and genoprob calculation
library(qtl2)

# read data
iron <- read_cross2(system.file("extdata", "iron.zip", package="qtl2"))

# insert pseudomarkers into map
map <- insert_pseudomarkers(iron$gmap, step=1)

# calculate genotype probabilities
probs <- calc_genoprob(iron, map, error_prob=0.002)

# inferred genotype at a 28.6 cM on chr 16
geno <- maxmarg(probs, map, chr=16, pos=28.6, return_char=TRUE)

# plot phenotype vs genotype
ggplot_pxg(geno, log10(iron$pheno[,1]), ylab=expression(log[10](Liver)))

# include +/- 2 SE intervals
ggplot_pxg(geno, log10(iron$pheno[,1]), ylab=expression(log[10](Liver)), SEmult=2)

# plot just the means
ggplot_pxg(geno, log10(iron$pheno[,1]), ylab=expression(log[10](Liver)), omit_points=TRUE)

# plot just the means +/- 2 SEs
ggplot_scan1

ggplot_pxs(geno, log10(iron$pheno[,1]), ylab=expression(log[10](Liver)), omit_points=TRUE, SEmul=2)

---

ggplot_scan1  

Plot a genome scan

Description

Plot LOD curves for a genome scan

Usage

ggplot_scan1(
x,  
map,  
lodcolumn = 1,  
chr = NULL,  
gap = 25,  
bcolor = "gray90",  
altbcolor = "gray85",  
...  
)

# S3 method for class 'scan1'
autoplot(
x,  
map,  
lodcolumn = 1,  
chr = NULL,  
gap = 25,  
bcolor = "gray90",  
altbcolor = "gray85",  
...  
)

ggplot_scan1_internal(
map,  
lod,  
gap = 25,  
col = NULL,  
shape = NULL,  
pattern = NULL,  
facet = NULL,  
patterns = c("none", "all", "hilit"),  
...  
)
Arguments

- **x**: Output of `scan1`.
- **map**: Map of pseudomarker locations.
- **lodcolumn**: LOD score column to plot (a numeric index, or a character string for a column name). One or more value(s) allowed.
- **chr**: Selected chromosomes to plot; a vector of character strings.
- **gap**: Gap between chromosomes.
- **bgcolor**: Background color for the plot.
- **altbgcolor**: Background color for alternate chromosomes.
- **...**: Additional graphics parameters.
- **lod**: Matrix of lod (or other) values.
- **col**: Colors for points or lines, with labels.
- **shape**: Shapes for points.
- **pattern**: Use to group values for plotting (default = `NULL`); typically provided by `plot_snpasso` internal routine.
- **facet**: Plot facets if multiple phenotypes and group provided (default = `NULL`).
- **patterns**: Connect SDP patterns: one of `c("none","all","hilit")`.

Value

None.

See Also

`ggplot_coef`, `ggplot_snpasso`

Examples

# load qtl2 package for data and genoprob calculation
library(qtl2)

# read data
iron <- read_cross2(system.file("extdata", "iron.zip", package="qtl2"))

# insert pseudomarkers into map
map <- insert_pseudomarkers(iron$gmap, step=1)

# calculate genotype probabilities
probs <- calc_genoprob(iron, map, error_prob=0.002)

# grab phenotypes and covariates; ensure that covariates have names attribute
pheno <- iron$pheno
covar <- match(iron$covar$sex, c("f", "m")) # make numeric
names(covar) <- rownames(iron$covar)
Xcovar <- get_x_covar(iron)
# perform genome scan
out <- scan1(probs, pheno, addcovar=covar, Xcovar=Xcovar)

# plot the results for selected chromosomes
chr <- c(2,7,8,9,15,16)
ggplot_scan1(out, map, lodcolumn=1:2, chr=chr, col=c("darkslateblue","violetred"),
legend.position=c(0.1,0.9))

# plot just one chromosome
ggplot_scan1(out, map, chr=8, lodcolumn=1:2, col=c("darkblue","violetred"))

# can also use autoplot from ggplot2
# lodcolumn can also be a column name
library(ggplot2)
autoplot(out, map, chr=8, lodcolumn=c("liver","spleen"), col=c("darkblue","violetred"))

###

#### ggplot_snpasso

**Plot SNP associations**

**Description**

Plot SNP associations, with possible expansion from distinct snps to all snps.

**Usage**

```r
ggplot_snpasso(
  scan1output,
  snpinfo,
  genes = NULL,
  lodcolumn = 1,
  show_all_snps = TRUE,
  drop_hilit = NA,
  col_hilit = "violetred",
  col = "darkslateblue",
  ylim = NULL,
  gap = 25,
  minlod = 0,
  bgcolor = "gray90",
  altbgcolor = "gray85",
  ...
)
```

**Arguments**

- `scan1output`: Output of `scan1`. Should contain an attribute, "snpinfo", as when `scan1` are run with SNP probabilities produced by `genoprob_to_snpprob`.
- `snpinfo`: Data frame with SNP information with the following columns (the last three are generally derived from with `index_snps`):
ggplot_snpasso

• **chr** - Character string or factor with chromosome
• **pos** - Position (in same units as in the "map" attribute in genoprobs).
• **sdp** - Strain distribution pattern: an integer, between 1 and $2^n - 2$ where $n$ is the number of strains, whose binary encoding indicates the founder genotypes
• **snp** - Character string with SNP identifier (if missing, the rownames are used).
• **index** - Indices that indicate equivalent groups of SNPs.
• **intervals** - Indexes that indicate which marker intervals the SNPs reside.
• **on_map** - Indicate whether SNP coincides with a marker in the genoprobs
genes Optional data frame containing gene information for the region, with columns 'start' and 'stop' in Mbp, 'strand' (as '"-"', '"+"', or 'NA'), and 'Name'. If included, a two-panel plot is produced, with SNP associations above and gene locations below.
lodcolumn LOD score column to plot (a numeric index, or a character string for a column name). One or more value(s) allowed.
show_all_snps If TRUE, expand to show all SNPs.
drop_hilit SNPs with LOD score within this amount of the maximum SNP association will be highlighted.
col_hilit Color of highlighted points
col Color of other points ylim y-axis limits
gap Gap between chromosomes.
minlod Minimum LOD to display. (Mostly for GWAS, in which case using ‘minlod=1’ will greatly increase the plotting speed, since the vast majority of points would be omitted.
bgcolor Background color for the plot.
altdbgcolor Background color for alternate chromosomes.
... Additional graphics parameters.

**Value**

object of class **ggplot**.

**Hidden graphics parameters**

A number of graphics parameters can be passed via ‘...’. For example, ‘bgcolor’ to control the background color and ‘altdbgcolor’ to control the background color on alternate chromosomes. ‘cex’ for character expansion for the points (default 0.5), ‘pch’ for the plotting character for the points (default 16), and ‘ylim’ for y-axis limits.

**See Also**

**ggplot_scan1, ggplot_coef**
Examples

```r
# Not run:
## load example DO data from web
library(qtl2)
file <- paste0("https://raw.githubusercontent.com/rqtl/",
  "qtl2data/master/DOex/DOex.zip")
DOex <- read_cross2(file)

# subset to chr 2
DOex <- DOex[,2]

# calculate genotype probabilities and convert to allele probabilities
pr <- calc_genoprob(DOex, error_prob=0.002)
apr <- genoprob_to_alleleprob(pr)

# query function for grabbing info about variants in region
snp_dbfile <- system.file("extdata", "cc_variants_small.sqlite", package="qtl2")
query_variants <- create_variant_query_func(snp_dbfile)

# SNP association scan
out_snps <- scan1snps(apr, DOex$pmap, DOex$pheno, query_func=query_variants,
  chr=2, start=97, end=98, keep_all_snps=TRUE)

# plot results
ggplot_snpasso(out_snps, snpinfo)

# can also just type autoplot() if ggplot2 attached
library(ggplot2)
autoplot(out_snps, snpinfo)

# plot just subset of distinct SNPs
autoplot(out_snps, snpinfo, show_all_snps=FALSE)

# highlight the top snps (with LOD within 1.5 of max)
autoplot(out_snps, snpinfo, drop_hilit=1.5)

# highlight SDP patterns in SNPs; connect with lines.
autoplot(out_snps, snpinfo, patterns="all",drop_hilit=4)

# highlight top SDP patterns in SNPs; connect with lines.
autoplot(out_snps, snpinfo, patterns="hilit",drop_hilit=4)

# query function for finding genes in region
gene_dbfile <- system.file("extdata", "mouse_genes_small.sqlite", package="qtl2")
query_genes <- create_gene_query_func(gene_dbfile)
genes <- query_genes(2, 97, 98)

# plot SNP association results with gene locations
autoplot(out_snps$lod, out_snps$snpinfo, drop_hilit=1.5, genes=genes)
```

## End(Not run)
Index

* hgraphics
  ggplot_genes, 6
  autoplot.genes (ggplot_genes), 6
  autoplot.scan1 (ggplot_scan1), 13
  autoplot.scan1coef (ggplot_coef), 4

CCcolors, 5
  color_patterns_get, 2
  color_patterns_pheno, 2
  color_patterns_set, 3

facet_wrap, 3
find_peaks, 9, 10

genoprob_to_snpprob, 15
  ggplot, 5, 7, 8, 12, 16
  ggplot_coef, 4, 14, 16
  ggplot_coefCC (ggplot_coef), 4
  ggplot_genes, 6
  ggplot_genes_internal, 7
  ggplot_onegeno, 8
  ggplot_peaks, 9
  ggplot_prg, 10
  ggplot_scan1, 5, 13, 16
  ggplot_scan1_internal (ggplot_scan1), 13
  ggplot_snpasso, 5, 14, 15
  guess_phase, 8

index_snps, 15
insert_pseudomarkers, 4

maxmarg, 8, 11
mean_prg (ggplot_prg), 10

plot, 6, 12
plot_coef, 12
plot_snpasso, 14

scan1, 3, 14, 15
scan1coef, 4