# Package ‘qtl2ggplot’

**March 11, 2023**

**Version** 1.2.2  
**Date** 2023-03-11  
**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, ggrepel  
**Suggests** devtools, testthat, roxygen2, knitr, rmarkdown  
**VignetteBuilder** knitr  
**License** GPL-3  
**URL** [https://github.com/byandell/qtl2ggplot](https://github.com/byandell/qtl2ggplot), [https://kbroman.org/qtl2/](https://kbroman.org/qtl2/)  
**BugReports** [https://github.com/byandell/qtl2ggplot/issues](https://github.com/byandell/qtl2ggplot/issues)  
**Encoding** UTF-8  
**ByteCompile** true  
**LinkingTo** Rcpp  
**RoxygenNote** 7.2.3  
**NeedsCompilation** yes  
**Author** Brian S Yandell [aut, cre], Karl W Broman [aut]  
**Maintainer** Brian S Yandell <brian.yandell@wisc.edu>  
**Repository** CRAN  
**Date/Publication** 2023-03-11 22:50:03 UTC
color_patterns_get

Set up col, pattern and group for plotting.

Description

Set up col, pattern and group for plotting.

Usage

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)

Value

list of colors and shapes.
**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`: 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.

---

**color_patterns_set**  
Set up colors for patterns or points

**Description**
Set up colors for patterns or points
Usage

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

Value

  list of col and pattern.

---

**ggplot_coef**  
*Plot QTL effects along chromosome*

Description

Plot estimated QTL effects along a chromosomes.

Usage

  ggplot_coef(
    object,
    map,
    columns = NULL,
    col = NULL,
ggplot_coef

```r
scan1_output = NULL,
gap = 25,
ylim = NULL,
bgcolor = "gray90",
altbgcolor = "gray85",
ylab = "QTL effects",
xlim = NULL,
...
)

ggplot_coefCC(object, map, colors = qtl2::CCcolors, ...)

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

**Arguments**

- `object` 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.
- `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

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

# insert pseudomarkers into map
map <- qtl2::insert_pseudomarkers(iron$gmap, step=1)

# calculate genotype probabilities
probs <- qtl2::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 <- qtl2::scan1coef(probs[,7], pheno, addcovar=covar)

# plot QTL effects
ggplot2::autoplot(coef, map[7], columns=1:3)
```

---

ggplot_genes  
Plot gene locations for a genomic interval

Description

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

Usage

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

## S3 method for class 'genes'

```r
autoplot(object, ...)
```

Arguments

- **object**: Object of class object
- **xlim**: x-axis limits (in Mbp)
**Description**

Plot genes at positions

**Usage**

```r
ggplot_genes_internal(
  start, 
  end, 
  strand, 
  rect_top, 
  rect_bottom, 
  colors, 
  space, 
  y, 
  dir_symbol, 
  name, 
  xlim, 
  xlab = "Position (Mb)", 
  ylab = "", 
  bgcolor = "gray92", 
  xat = NULL, 
)```
ggplot_listof_scan1coef

Plot object of class listof_scan1coeff, which is a list of objects of class scan1coef.

Usage

ggplot_listof_scan1coef(
  object,
  map,
  columns = NULL,
  col = NULL,
  scan1_output = NULL,
  facet = "pattern",
  ...
)

## S3 method for class 'listof_scan1coeff'
autoplot(object, ...)

Arguments

object object of class listof_scan1coeff
map A list of vectors of marker positions, as produced by insert_pseudomarkers.
columns Vector of columns to plot
ggplot_onegeno

- **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.
- **facet**: Plot facets if multiple phenotypes and group provided (default = "pattern").
- **pattern**: Use phenotype names as pattern.

**Value**

Object of class `ggplot`.

**Author(s)**

Brian S Yandell, <brian.yandell@wisc.edu>

---

**Description**

Plot one individual’s genome-wide genotypes

**Usage**

```r
ggplot_onegeno(
geno,
map,
ind = 1,
chr = NULL,
col = NULL,
shift = FALSE,
chrwidth = 0.5,
...
)
```

**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 (can be a vector).
- **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

# 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$map, step=1)

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

# inferred genotypes
geno <- maxmarg(probs)

# plot the inferred genotypes for the first individual
ggplot_onegeno(geno, map, shift = TRUE)

# plot the inferred genotypes for the first four individuals
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

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

    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

Value

None.

See Also

find_peaks

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)

# find peaks above lod=3.5 (and calculate 1.5-LOD support intervals)
peaks <- find_peaks(out, map, threshold=3.5, drop=1.5)

# color peaks above 6 red; only show chromosomes with peaks
plot_peaks(peaks, map)
peaks$col <- (peaks$lod > 6)

ggplot_peaks(peaks, map[names(map) %in% peaks$chr], col = c("blue","red"),
            legend.title = "LOD > 6")

---

**ggplot_pxg**

*Plot phenotype vs genotype*

**Description**

Plot phenotype vs genotype for a single putative QTL and a single phenotype.

**Usage**

```r
ggplot_pxg(
geno,
pheno,
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,
...)
```

```r
mean_pxg(geno, pheno, dataframe = NULL)
```

**Arguments**

- `geno` Vector of genotypes, as produced by `maxmarg` with specific `chr` and `pos`.
- `pheno` 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_pxg(geno, log10(iron$pheno[,1]), ylab=expression(log[10](Liver)),
           omit_points=TRUE, SEmult=2)

---

ggplot_scan1  

*Plot a genome scan*

**Description**

Plot LOD curves for a genome scan

**Usage**

```
ggplot_scan1(
  object,
  map,
  lodcolumn = 1,
  chr = NULL,
  gap = 25,
  bgcolor = "gray90",
  altbgcolor = "gray85",
  ...
)
```

## S3 method for class 'scan1'

```
autoplot(object, ...)
```

```
 ggplot_scan1_internal(
   map,
   
```
ggplot_scan1

lod,
gap = 25,
col = NULL,
shape = NULL,
pattern = NULL,
facet = NULL,
patterns = c("none", "all", "hilit"),
chrName = "Chr",
...)

Arguments

object 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").

chrName Add prefix chromosome name (default "Chr").

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)
autplot(out, map, chr=8, lodcolumn=c("liver","spleen"), col=c("darkblue","violetred"))
```r

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`):
  • 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  y-axis limits
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.
altbgcolor  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 ‘altbgcolor’ 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

dirpath <- "https://raw.githubusercontent.com/rqtl/qtl2data/master/DOex"

# Read DOex example cross from 'qt12data'
DOex <- subset(qtl2::read_cross2(file.path(dirpath, "DOex.zip")), chr = "2")

# Download genotype probabilities
tmpfile <- tempfile()
download.file(file.path(dirpath, "DOex_genoprobs_2.rds"), tmpfile, quiet=TRUE)
pr <- readRDS(tmpfile)
unlink(tmpfile)

# Download SNP info for DOex from web and read as RDS.
tmpfile <- tempfile()
download.file(file.path(dirpath, "c2_snpinfo.rds"), tmpfile, quiet=TRUE)
snpinfo <- readRDS(tmpfile)
unlink(tmpfile)

# Convert to SNP probabilities
snpinfo <- qtl2::index_snps(DOex$pmap, snpinfo)

# Scan SNPs.
scan_snppr <- qtl2::scan1(snpinfo, DOex$pheno)

# plot results
ggplot_snpasso(scan_snppr, snpinfo, show_all_snps=FALSE, patterns="all", drop_hilit=1.5)

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

# plot just subset of distinct SNPs
autoplot(scan_snppr, snpinfo, show_all_snps=FALSE, drop_hilit=1.5)

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

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

# plot SNP association results with gene locations
autoplot(scan_snppr, snpinfo, patterns="hilit", drop_hilit=1.5, genes=genes)
```

---

### listof_scan1coef

**List of scan1coef objects**

#### Description

Create a list of scan1coef objects using `scan1coef`.

Summary of object of class `listof_scan1coef`, which is a list of objects of class `scan1coef`.

Summary of object of class `listof_scan1coef`, which is a list of objects of class `scan1coef`.

Subset of object of class `listof_scan1coef`, which is a list of objects of class `scan1coef`.

#### Usage

```r
listof_scan1coef(
  probs,
  phe,
  K = NULL,
  covar = NULL,
  blups = FALSE,
  center = FALSE,
  ...
)

summary_listof_scan1coef(
  object,
  scan1_object,
  map,
  coef_names = dimnames(object[[1]])[[2]],
  center = TRUE,
  ...
)
```

```r
summary_scan1coef(object, scan1_object, map, ...)
```
## S3 method for class 'scan1coef'
summary(object, ...)

subset_listof_scan1coef(x, elements, ...)

## S3 method for class 'listof_scan1coef'
subset(x, ...)

## S3 method for class 'listof_scan1coef'
x[...]

### Arguments
- **probs**: genotype probabilities object for one chromosome from `calc_genoprob`
- **phe**: data frame of phenotypes
- **K**: list of length 1 with kinship matrix
- **covar**: matrix of covariates
- **blups**: Create BLUPs if TRUE
- **center**: center coefficients if TRUE
- **...**: ignored
- **object**: object of class `listof_scan1coef`
- **scan1_object**: object from `scan1`
- **map**: A list of vectors of marker positions, as produced by `insert_pseudomarkers`.
- **coef_names**: names of effect coefficients (default is all coefficient names)
- **x**: object of class `listof_scan1coef`
- **elements**: indexes or names of list elements in x

### Value
object of class `listof_scan1coef`

### Author(s)
Brian S Yandell, <brian.yandell@wisc.edu>

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

# insert pseudomarkers into map
map <- qtl2::insert_pseudomarkers(iron$gmap, step=1)

# calculate genotype probabilities
probs <- qtl2::calc_genoprob(iron, map, error_prob=0.002)
```
# Ensure that covariates have names attribute
covar <- match(iron$covar$sex, c("f", "m")) # make numeric
names(covar) <- rownames(iron$covar)

# Calculate scan1coef on all phenotypes,
# returning a list of \code{\link{scan1coef}} objects
out <- listof_scan1_coef(probs[,7], iron$pheno, addcovar = covar, center = TRUE)

# Plot coefficients for all phenotypes
ggplot2::autoplot(out, map[7], columns = 1:3)

# Summary of coefficients at scan peak
scan_pr <- qtl2::scan1(probs[,7], iron$pheno)
summary(out, scan_pr, map[7])

# sdptopattern

sdptopattern  Convert sdptopattern

Description

Convert strain distribution pattern (sdp) to letter pattern. Taken from package ‘qtl2pattern’ for internal use here.

Usage

sdptopattern(sdp, haplos, symmetric = TRUE)

Arguments

sdp  vector of sdp values
haplos  letter codes for haplotypes (required)
symmetric  make patterns symmetric if TRUE

Value

vector of letter patterns

Author(s)

Brian S Yandell, <brian.yandell@wisc.edu>
summary_scan1  Summary of scan1 object

Description
Summary of scan1 object

Usage
summary_scan1(
  object,
  map,
  snpinfo = NULL,
  lodcolumn = seq_len(ncol(object)),
  chr = names(map),
  sum_type = c("common", "best"),
  drop = 1.5,
  show_all_snps = TRUE,
  ...
)

## S3 method for class 'scan1'
summary(object, ...)

Arguments

object  object from scan1
map     A list of vectors of marker positions, as produced by insert_pseudomarkers.
snpinfo Data frame with SNP information with the following columns (the last three are generally derived from with index_snps):
- chr - Character string or factor with chromosome
- pos - Position (in same units as in the "map" attribute in genoprobs.
- snpd - 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

lodcolumn one or more lod columns
chr one or more chromosome IDs
sum_type type of summary
drop LOD drop from maximum
show_all_snps show all SNPs if TRUE
... other arguments not used
Value

tbl summary

Author(s)

Brian S Yandell, <brian.yandell@wisc.edu>

Examples

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

# insert pseudomarkers into map
map <- qtl2::insert_pseudomarkers(iron$gmap, step=1)

# calculate genotype probabilities
probs <- qtl2::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 <- qtl2::get_x_covar(iron)

# perform genome scan
out <- qtl2::scan1(probs, pheno, addcovar=covar, Xcovar=Xcovar)

# summary
summary(out, map)
Index

* **hgraphics**
  - `ggplot_genes`, 6
* **hplot**
  - `ggplot_listof_scan1coef`, 8
* **utilities**
  - `listof_scan1coef`, 19
  - `sdp_to_pattern`, 21
  - `summary_scan1`, 22
  - `.listof_scan1coef(listof_scan1coef)`, 19

  - `autoplot.genes(ggplot_genes)`, 6
  - `autoplot.listof_scan1coef(ggplot_listof_scan1coef)`, 8
  - `autoplot.scan1(ggplot_scan1)`, 14
  - `autoplot.scan1coef(ggplot_coef)`, 4

  - `calc_genoprob`, 20
  - `CCcolors`, 5
  - `color_patterns_get`, 2
  - `color_patterns_pheno`, 3
  - `color_patterns_set`, 3

  - `facet_wrap`, 3
  - `find_peaks`, 11

  - `genoprob_to_snpprob`, 17
  - `ggplot`, 5, 8–10, 13, 18
  - `ggplot_coef`, 4, 9, 15, 18
  - `ggplot_coefCC(ggplot_coef)`, 4
  - `ggplot_genes`, 6
  - `ggplot_genes_internal`, 7
  - `ggplot_listof_scan1coef`, 8
  - `ggplot_onegeno`, 9
  - `ggplot_peaks`, 10
  - `ggplot_pxg`, 12
  - `ggplot_scan1`, 5, 14, 18
  - `ggplot_scan1_internal(ggplot_scan1)`, 14
  - `ggplot_snpasso`, 5, 15, 16
  - `guess_phase`, 9

  - `index_snps`, 17, 22
  - `insert_pseudomarkers`, 5, 8, 20, 22

  - `listof_scan1coef`, 19, 19
  - `maxmarg`, 9, 12
  - `mean_pxg(ggplot_pxg)`, 12
  - `plot`, 7, 13
  - `plot_coef`, 13
  - `plot_snpasso`, 15

  - `scan1`, 4, 15, 17, 22
  - `scan1coef`, 5, 19
  - `sdp_to_pattern`, 21
  - `subset.listof_scan1coef(listof_scan1coef)`, 19
  - `subset_listof_scan1coef(listof_scan1coef)`, 19
  - `summary.listof_scan1coef(listof_scan1coef)`, 19
  - `summary.scan1(summary_scan1)`, 22
  - `summary.scan1coef(listof_scan1coef)`, 19
  - `summary_listof_scan1coef(listof_scan1coef)`, 19
  - `summary_scan1`, 22
  - `summary_scan1coef(listof_scan1coef)`, 19

24