Package 'qtl2pattern'

March 11, 2023

Type Package

Title Pattern Support for 'qtl2' Package

Version 1.2.1

Date 2023-03-11

Description Routines in 'qtl2' to study allele patterns in quantitative trait loci (QTL) mapping over a chromosome.
Useful in crosses with more than two alleles to identify how sets of alleles, genetically different strands at the same locus, have different response levels.
Plots show profiles over a chromosome.
Can handle multiple traits together.
See <https://github.com/byandell/qtl2pattern>.

Depends R (>= 3.1.0)

Imports dplyr, tidyr, stringr, ggplot2, assertthat, qtl2, qtl2fst, fst, rlang, stats, graphics

Suggests knitr, rmarkdown, qtl2ggplot

VignetteBuilder knitr

License GPL-3

URL https://github.com/byandell/qtl2pattern

Encoding UTF-8

RoxygenNote 7.2.3

NeedsCompilation no

Author Brian S Yandell [aut, cre]

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

Repository CRAN

Date/Publication 2023-03-11 22:30:02 UTC

R topics documented:

allele1 2
create_probs_query_func 3

gene_exon 5
genoprob_to_patternprob 7
get.gene.locations 8
get_feature_snp 9
get_gene_snp 10
ggplot_merge_feature 10
ggplot_scan1pattern 12
pattern_diplos 15
pattern_label 15
read_fast 16
read_probs 17
sdp_to_pattern 18
snpinfo_to_map 19
snpprob_collapse 19
summary.feature_snp 20
summary.feature_tbl 21
summary.gene_snp 23
top_snps_pattern 23

Index **26**

`allele1` *Allele plot for SNPs, alleles and allele pairs*

Description

Create table of alleles for various model fits.

Plot alleles for haplotype, diplotype and top patterns and genome position.

Usage

```
allele1(
  probD,
  phe_df = NULL,
  cov_mx = NULL,
  map = NULL,
  K_chr = NULL,
  patterns = NULL,
  alt = NULL,
  blups = FALSE,
  ...
)

ggplot_allele1(
  object,
  scan1_object = NULL,
  map = NULL,
```

```
  pos = NULL,
  trim = TRUE,
  legend.position = "none",
  ...
)

## S3 method for class 'allele1'
autoplot(object, ...)
```

Arguments

probD	object of class calc_genoprob
phe_df	data frame with one phenotype
cov_mx	covariate matrix
map	Genome map (required if scan1_object present).
K_chr	kinship matrix
patterns	data frame of pattern information
alt	Haplotype allele letter(s) for alternative to reference.
blups	Create BLUPs if TRUE
...	Other parameters ignored.
object	Object of class allele1.
scan1_object	Optional object of class scan1 to find peak.
pos	Genome position in Mbp (supercedes scan1_object).
trim	If TRUE, trim extreme alleles.
legend.position	Legend position (default is "none").

Value

Table with allele effects across sources.

object of class ggplot

---

create_probs_query_func

Create a function to query genotype probabilities

Description

Create a function that will connect to a database of genotype probability information and return a list with ‘probs‘ object and a ‘map‘ object.

Usage

```
create_probs_query_func(dbfile, method_val = "fst", probdir_val = "genoprob")
```

Arguments

dbfile	Name of database file
method_val	either "fst" or "calc" for type of genotype probabilities
probdir_val	name of probability directory (default "genoprob")

Details

Note that this function assumes that probdir_val has a file with the physical map with positions in Mbp and other files with genotype probabilities. See read_probs for details on how probabilities are read. See create_variant_query_func for original idea.

Value

Function with six arguments, 'chr', 'start', 'end', 'allele', 'method' and 'probdir'. It returns a list with 'probs' and 'map' objects spanning the region specified by the first three arguments. The 'probs' element should be either a 'calc_genoprob' or 'fst_genoprob' object (see fst_genoprob).

Examples

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

create_qv <- function(dirpath) {
  # Download SNP info for DOex from web via RDS.
  # snpinfo is referenced internally in the created function.

  tmpfile <- tempfile()
  download.file(file.path(dirpath, "c2_snpinfo.rds"), tmpfile, quiet=TRUE)
  snpinfo <- readRDS(tmpfile)
  unlink(tmpfile)
  snpinfo <- dplyr::rename(snpinfo, pos = pos_Mbp)

  function(chr, start, end) {
    if(chr != "2") return(NULL)
    if(start < 96.5) start <- 96.5
    if(end > 98.5) end <- 98.5
    if(start >= end) return(NULL)
    dplyr::filter(snpinfo, .data$pos >= start, .data$pos <= end)
  }
}

query_variants <- create_qv(dirpath)


create_qg <- function(dirpath) {
  # Download Gene info for DOex from web via RDS
  # gene_tbl is referenced internally in the created function.

  tmpfile <- tempfile()
  download.file(file.path(dirpath, "c2_genes.rds"), tmpfile, quiet=TRUE)
  gene_tbl <- readRDS(tmpfile)
  unlink(tmpfile)
```

```
  function(chr, start, end) {
    if(chr != "2") return(NULL)
    if(start < 96.5) start <- 96.5
    if(end > 98.5) end <- 98.5
    if(start >= end) return(NULL)
    dplyr::filter(gene_tbl, .data$end >= start, .data$start <= end)
  }
}

query_genes <- create_qg(dirpath)


# Examples for probs require either FST or RDS storage of data.
```

---

gene_exon *Get exons for set of genes*

Description

Match up exon start,stop,strand with genes. Use query_genes to find features; see create_gene_query_func.

Returns table of gene and its exons.

Uses gene_exon to plot genes, exons, mRNA with SNPs.

Usage

```
gene_exon(
  top_snps_tbl,
  feature_tbl = query_genes(chr_id, range_Mbp[1], range_Mbp[2])
)

## S3 method for class 'gene_exon'
summary(object, gene_name = NULL, top_snps_tbl = NULL, extra = 0.005, ...)

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

ggplot_gene_exon(
  object,
  top_snps_tbl = NULL,
  plot_now = TRUE,
  genes = unique(object$gene),
  ...
)

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

Arguments

top_snps_tbl	table from top_snps
feature_tbl	table of features from query_genes; see create_gene_query_func
object	Object of class gene_exon.
gene_name	name of gene as character string
extra	extra region beyond gene for SNPs (in Mbp)
...	arguments passed along to gene_exon
x	Object of class gene_exon.
gene_val	Name of gene from object x.
plot_now	plot now if TRUE
genes	Names of genes in object

Value

tbl of exon and gene features

tbl of summary

list of ggplots (see gene_exon)

Author(s)

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

Examples

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

# Read DOex example cross from 'qtl2data'
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)
snpinfo <- dplyr::rename(snpinfo, pos = pos_Mbp)

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

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

# Collect top SNPs
top_snps_tbl <- top_snps_pattern(scan_snppr, snpinfo)

# Download Gene info for DOex from web via RDS
tmpfile <- tempfile()
download.file(file.path(dirpath, "c2_genes.rds"), tmpfile, quiet=TRUE)
gene_tbl <- readRDS(tmpfile)
unlink(tmpfile)

# Get Gene exon information.
out <- gene_exon(top_snps_tbl, gene_tbl)
summary(out, gene = out$gene[1])
```

`genoprob_to_patternprob` *Collapse genoprob according to pattern*

Description

Collapse genoprob according to pattern

Usage

```
genoprob_to_patternprob(probs1, sdp, alleles = FALSE)
```

Arguments

`probs1`	object of class `calc_genoprob`
`sdp`	SNP distribution pattern
`alleles`	use allele string if `TRUE`

Value

object of class `calc_genoprob`

Author(s)

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

Examples

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

# Read DOex example cross from 'qtl2data'
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)

# Convert genotype probabilities to pattern probabilities for pattern 1.
pattern_pr <- genoprob_to_patternprob(pr, 7, TRUE)

str(pr)
str(pattern_pr)
```

`get.gene.locations` *Helper function to set gene locations on plot.*

Description

Figure out gene locations to make room for gene names. Written original by Dan Gatti 2013-02-13

Usage

```
get.gene.locations(
  locs,
  xlim,
  text_size = 3,
  str_rect = c("iW", "i"),
  n_rows = 10,
  plot_width = 6,
  ...
)
```

Arguments

`locs`	tbl of gene information
`xlim`	X axis limits
`text_size`	size of text (default 3)
`str_rect`	character spacing on left and right of rectangles (default c("iW","i"))
`n_rows`	desired number of rows (default 10)

plot_width	width of default plot window (in inches)
...	additional parameters (not used)

Value

list object used by ggplot_feature_tbl

Author(s)

Brian S Yandell, <brian.yandell@wisc.edu> Daniel Gatti, <Dan.Gatti@jax.org>

References

https://github.com/dmgatti/DOQTL/blob/master/R/gene.plot.R

---

get_feature_snp *Match features with SNPs*

Description

Find features that overlap with SNPs

Usage

```
get_feature_snp(snp_tbl, feature_tbl, extend = 0.005)
```

Arguments

snp_tbl	tbl of SNPs from assoc.map
feature_tbl	tbl of feature information from create_gene_query_func
extend	extend region for SNPs in Mbp (default 0.005)

Value

tbl of features covering SNPs

Author(s)

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

get_gene_snp *Match genes with SNPs*

Description

Internal routine to find features that overlap with SNPs

Usage

```
get_gene_snp(
  snp_tbl,
  feature_tbl,
  feature_snp = get_feature_snp(snp_tbl, feature_tbl, 0)
)
```

Arguments

snp_tbl	tbl of SNPs from query_variants; see package create_variant_query_func
feature_tbl	tbl of feature information from query_genes; see package create_gene_query_func
feature_snp	tbl of feature information from get_feature_snp

Value

tbl of genes covering SNPs

Author(s)

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

ggplot_merge_feature *Plot of merge_feature object*

Description

Merge all SNPs in small region with LOD peaks across multiple phenotype.

Usage

```
ggplot_merge_feature(object, pheno, plot_by = c("pattern", "consequence"), ...)

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

merge_feature(
  top_snps_tbl,
```

```
  snpinfo,
  out_lmm_snps,
  drop = 1.5,
  dropchar = 0,
  exons = gene_exon(top_snps_tbl)
)

## S3 method for class 'merge_feature'
summary(object, sum_type = c("SNP type", "pattern"), ...)
```

Arguments

object	of class merge_feature
pheno	name of phenotype to be plotted
plot_by	element to plot by (one of c("pattern","consequence"))
...	other arguments not used
top_snps_tbl	tbl from top_snps_pattern or top_snps
snpinfo	SNP information table
out_lmm_snps	tbl from scan1 on SNPs
drop	include LOD scores within drop of max for each phenotype
dropchar	number of characters to drop on phenames
exons	table from gene_exon
sum_type	one of c("SNP type","pattern")

Value

ggplot2 object

tbl with added information on genes and exons

table summary

Author(s)

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

Examples

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

# Read DOex example cross from 'qtl2data'
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)
snpinfo <- dplyr::rename(snpinfo, pos = pos_Mbp)

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

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

# Collect top SNPs
top_snps_tbl <- top_snps_pattern(scan_snppr, snpinfo)
summary(top_snps_tbl)

# Download Gene info for DOex from web via RDS
tmpfile <- tempfile()
download.file(file.path(dirpath, "c2_genes.rds"), tmpfile, quiet=TRUE)
gene_tbl <- readRDS(tmpfile)
unlink(tmpfile)

out <- merge_feature(top_snps_tbl, snpinfo, scan_snppr, exons = gene_tbl)
summary(out, "pattern")
```

`ggplot_scan1pattern` *Plot scan pattern usign ggplot2*

Description

Plot scan pattern usign ggplot2

Genome scan by pattern set

Usage

```
ggplot_scan1pattern(
  object,
  map,
  plot_type = c("lod", "coef", "coef_and_lod"),
  patterns = object$patterns$founders,
  columns = 1:3,
  min_lod = 3,
  lodcolumn = seq_along(patterns),
  facet = "pheno",
```

```
  ...
)

## S3 method for class 'scan1pattern'
autoplot(object, ...)

scan1pattern(
  probs1,
  phe,
  K = NULL,
  covar = NULL,
  map,
  patterns,
  condense_patterns = TRUE,
  blups = FALSE,
  do_scans = TRUE
)

## S3 method for class 'scan1pattern'
summary(object, map, ...)
```

Arguments

object	object of class scan1pattern
map	genome map
plot_type	type of plot from c("lod","coef")
patterns	data frame of pattern information
columns	columns used for coef plot
min_lod	minimum LOD peak for contrast to be retained
lodcolumn	columns used for scan1 plot (default all patterns)
facet	Plot facets if multiple phenotypes and patterns provided (default = "pheno").
...	additional parameters passed on to other methods
probs1	object of class calc_genoprob
phe	data frame with one phenotype
K	kinship matrix
covar	covariate matrix
condense_patterns	remove snp_action from contrasts if TRUE
blups	Create BLUPs if TRUE
do_scans	Do scans if TRUE.

Value

object of class ggplot

List containing:

- patterns Data frame of summary for top patterns (column `founders` has pattern)
- dip_set Diplotype sets for contrasts
- group Group for each founder pattern
- scan Object of class `scan1`.
- coef Object of class `listof_scan1coef`. See package 'qtl2ggplot'.

Author(s)

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

Examples

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

# Read DOex example cross from 'qtl2data'
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)
snpinfo <- dplyr::rename(snpinfo, pos = pos_Mbp)

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

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

# Summarize to find top patterns
patterns <- dplyr::arrange(summary(top_snps_tbl), dplyr::desc(max_lod))

# Scan using patterns.
scan_pat <- scan1pattern(pr, DOex$pheno, map = DOex$pmap, patterns = patterns)

# Summary of scan1pattern.
summary(scan_pat, DOex$pmap)
```

pattern_diplos *Extract pattern of diplotypes*

Description

Extract pattern of diplotypes

Extract pattern of haplotypes

Usage

```
pattern_diplos(sdp, haplos, diplos, cont = NULL)

pattern_haplos(sdp, haplos)
```

Arguments

sdp	vector of sdp from top_snps_pattern
haplos	vector of haplotype names
diplos	vector of diplotype names
cont	vector of types of contrasts (NULL or from c("add","dom","b6r","b6d"))

Value

matrix of diplotype patterns

matrix of haplotype patterns

Author(s)

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

pattern_label *Turn genotype probabilities into labels*

Description

Turn genotype probabilities into labels

Usage

```
pattern_label(genos, allele = TRUE)

pattern_sdp(label, sdp = NULL, geno_names = sort(unique(label)))
```

Arguments

genos	matrix of genotype probabilities at locus
allele	Driver has alleles if TRUE, otherwise allele pairs.
label	character string from pattern_label
sdp	SNP distribution pattern for plot colors
geno_names	unique genotype names (alleles or allele pairs)

Value

character vector of genotype names.

---

read_fast *Read fast database with possible rownames*

Description

Read fast database with format fst. Use first column of database (must be named 'ind') as rownames if desired. R/qtl2 routines assume data frames have rownames to use to align individuals.

Usage

```
read_fast(datapath, columns = NULL, rownames = TRUE)
```

Arguments

datapath	character string path to database
columns	names or indexes for columns to be extracted
rownames	use first column of rownames if TRUE (can supply column number)

Value

extracted data frame with appropriate rows and columns.

See Also

read_fst

`read_probs` *Read genotype probability object from file*

Description

Read object from file stored according to method.

Usage

```
read_probs(
  chr = NULL,
  start_val = NULL,
  end_val = NULL,
  datapath,
  allele = TRUE,
  method,
  probdir = "genoprob"
)
```

Arguments

`chr`	vector of chromosome identifiers
`start_val, end_val`	start and end values in Mbp
`datapath`	name of folder with Derived Data
`allele`	read haplotype allele probabilities (if `TRUE`) or diplotype allele-pair probabilities (if `FALSE`)
`method`	method of genoprob storage
`probdir`	genotype probability directory (default `"genoprob"`)

Value

list with `probs` = large object of class `calc_genoprob` and `map` = physical map for selected `chr`

Author(s)

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

sdp_to_pattern *Convert sdp to pattern*

Description

Convert strain distribution pattern (sdp) to letter pattern.

Usage

```
sdp_to_pattern(sdp, haplos, symmetric = TRUE)

sdp_to_logical(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>

Examples

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


# 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)
snpinfo <- dplyr::rename(snpinfo, pos = pos_Mbp)

# Extract strain distribution pattern.
sdp <- snpinfo$sdp
# Find out how many alleles.
nallele <- ceiling(log2(max(sdp)))
out <- sdp_to_pattern(sdp, LETTERS[seq_len(nallele)])
# Show most frequent patterns.
head(rev(sort(c(table(out)))))
```

snpinfo_to_map *Convert SNP info to map*

Description

Convert SNP info to map

Usage

```
snpinfo_to_map(snpinfo)
```

Arguments

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

Value

map as list of vectors of marker positions.

snpprob_collapse *Collapse genoprob according to pattern*

Description

Collapse genoprob according to pattern

Usage

```
snpprob_collapse(
  snpprobs,
  action = c("additive", "add+dom", "non-add", "recessive", "dominant", "basic")
)
```

Arguments

snpprobs	object of class calc_genoprob
action	SNP gene action type

Value

object of class calc_genoprob

Author(s)

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

Examples

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

# Read DOex example cross from 'qtl2data'
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)
snpinfo <- dplyr::rename(snpinfo, pos = pos_Mbp)

# Convert to snp probabilities
snpinfo <- qtl2::index_snps(DOex$pmap, snpinfo)
snppr <- qtl2::genoprob_to_snpprob(pr, snpinfo)


dim(snppr[[1]])
dim(snpprob_collapse(snppr, "additive")[[1]])
```

---

summary.feature_snp *Summary of features with SNP information*

Description

Summary of features with SNP information

Usage

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

Arguments

object	tbl of feature information from get_feature_snp
...	additional parameters ignored

Value

tbl of feature summaries by type

Author(s)

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

---

summary.feature_tbl *Summary of features*

Description

Show count min and max of features by type

Plot genes as rectangles followed by names. Stagger genes for easy reading. Written original by Dan Gatti 2013-02-13

Usage

```
## S3 method for class 'feature_tbl'
summary(object, major = TRUE, ...)

## S3 method for class 'feature_tbl'
subset(x, start_val = 0, stop_val = max(x$stop), ...)

ggplot_feature_tbl(
  object,
  rect_col = "grey70",
  strand_col = c(`-` = "#1b9e77", `+` = "#d95f02"),
  type_col = c(gene = "black", pseudogene = "#1b9e77", other = "#d95f02"),
  text_size = 3,
  xlim = NULL,
  snp_pos = top_snps_tbl$pos,
  snp_lod = top_snps_tbl$lod,
  top_snps_tbl = NULL,
  snp_col = "grey70",
  extend = 0.005,
```

```
  ...
)

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

Arguments

object	tbl of gene information from query_variants; see create_variant_query_func
major	if TRUE (default), only summarize genes and exons
...	additional arguments (not used)
x	tbl of feature information from get_feature_snp
start_val, stop_val	start and stop positions for subset
rect_col	fill color of rectangle (default "grey70")
strand_col	edge color of rectangle by strand from object (default -="blue", +="red"; none if NULL)
type_col	color of type from object (default "black" for gene, "blue" for pseudogene; none if NULL)
text_size	size of text (default 3)
xlim	horizontal axis limits (default is range of features)
snp_pos	position of SNPs in bp if used (default NULL)
snp_lod	LOD of SNPs (for color plotting)
top_snps_tbl	table from top_snps
snp_col	color of SNP vertical lines (default "grey70")
extend	extend region for SNPs in bp (default 0.005)

Value

tbl of feature summaries by type

tbl of feature summaries by type

data frame of gene information (invisible)

Author(s)

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

Brian S Yandell, <brian.yandell@wisc.edu> Daniel Gatti, <Dan.Gatti@jax.org>

References

https://github.com/dmgatti/DOQTL/blob/master/R/gene.plot.R

summary.gene_snp *Summary of genes overlapping SNPs*

Description

Summary of genes overlapping SNPs

Usage

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

Arguments

object	tbl of feature information from get_feature_snp
...	additional parameters ignored

Value

tbl of feature summaries by type

Author(s)

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

top_snps_pattern *Top SNPs organized by allele pattern*

Description

Separate fine mapping scans by allele pattern.

Usage

```
top_snps_pattern(
  scan1_output,
  snpinfo,
  drop = 1.5,
  show_all_snps = TRUE,
  haplos
)

## S3 method for class 'top_snps_pattern'
summary(object, sum_type = c("range", "best", "peak"), ...)

## S3 method for class 'top_snps_pattern'
subset(x, start_val = 0, end_val = max(x$pos), pheno = NULL, ...)
```

Arguments

scan1_output	output of linear mixed model for phename (see scan1)
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_id - 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
drop	include all SNPs within drop of max LOD (default 1.5)
show_all_snps	show all SNPs if TRUE
haplos	optional argument identify codes for haplotypes
object	object of class top_snps_tbl
sum_type	type of summary (one of "range","best")
...	additional parameters ignored
x	tbl of feature information from get_feature_snp
start_val, end_val	start and end positions for subset
pheno	phenotype name(s) for subset

Value

table of top_snps at maximum lod for pattern

table summary

subset of x

Author(s)

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

Examples

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

# Read DOex example cross from 'qtl2data'
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)
snpinfo <- dplyr::rename(snpinfo, pos = pos_Mbp)

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

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

# Collect top SNPs
top_snps_tbl <- top_snps_pattern(scan_snppr, snpinfo)
summary(top_snps_tbl)
```

Index

∗ **hplot**
gene_exon, 5
ggplot_merge_feature, 10
summary.feature_tbl, 21
∗ **utilities**
gene_exon, 5
genoprob_to_patternprob, 7
get.gene.locations, 8
get_feature_snp, 9
get_gene_snp, 10
ggplot_merge_feature, 10
ggplot_scan1pattern, 12
pattern_diplos, 15
read_probs, 17
sdp_to_pattern, 18
snpprob_collapse, 19
summary.feature_snp, 20
summary.feature_tbl, 21
summary.gene_snp, 23
top_snps_pattern, 23

allele1, 2, *3*
autoplot.allele1 *(*allele1*)*, 2
autoplot.feature_tbl
*(*summary.feature_tbl*)*, 21
autoplot.gene_exon *(*gene_exon*)*, 5
autoplot.merge_feature
*(*ggplot_merge_feature*)*, 10
autoplot.scan1pattern
*(*ggplot_scan1pattern*)*, 12

calc_genoprob, *3*, *7*, *13*, *17*, *20*
create_gene_query_func, *5, 6*, *9, 10*
create_probs_query_func, 3
create_variant_query_func, *4*, *10*, *22*

fst_genoprob, *4*

gene_exon, *5*, 5, *6*, *11*
genoprob_to_patternprob, 7
get.gene.locations, 8
get_feature_snp, 9, *10*, *21–24*
get_gene_snp, 10
ggplot, *3*, *13*
ggplot_allele1 *(*allele1*)*, 2
ggplot_feature_tbl, *9*
ggplot_feature_tbl
*(*summary.feature_tbl*)*, 21
ggplot_gene_exon *(*gene_exon*)*, 5
ggplot_merge_feature, 10
ggplot_scan1pattern, 12

index_snps, *19*, *24*

merge_feature *(*ggplot_merge_feature*)*, 10

pattern_diplos, 15
pattern_haplos *(*pattern_diplos*)*, 15
pattern_label, 15, *16*
pattern_sdp *(*pattern_label*)*, 15

read_fast, 16
read_fst, *16*
read_probs, *4*, 17

scan1, *3*, *11*, *14*, *24*
scan1pattern, *13*
scan1pattern *(*ggplot_scan1pattern*)*, 12
sdp_to_logical *(*sdp_to_pattern*)*, 18
sdp_to_pattern, 18
snpinfo_to_map, 19
snpprob_collapse, 19
subset.feature_tbl
*(*summary.feature_tbl*)*, 21
subset.gene_exon *(*gene_exon*)*, 5
subset.top_snps_pattern
*(*top_snps_pattern*)*, 23
summary.feature_snp, 20
summary.feature_tbl, 21
summary.gene_exon *(*gene_exon*)*, 5
summary.gene_snp, 23

summary.merge_feature
(ggplot_merge_feature), 10
summary.scan1pattern
(ggplot_scan1pattern), 12
summary.top_snps_pattern
(top_snps_pattern), 23

top_snps, *6*, *11*, *22*
top_snps_pattern, *11*, *15*, 23