Package ‘vqtl’

March 6, 2019

Title Genome Scans to Accommodate and Target Genetic and Non-Genetic Effects on Trait Variance in Test Crosses

Version 2.0.5

Description In recognition that there are many factors (genetic loci, macro-genetic factors such as sex, and environmental factors) that influence the extent of environmental variation, the ‘vqtl’ package conducts genome scans that accommodate and target these factors. The main functions of this package, scanonevar() and scanonevar.perm() take as input a cross object from the popular ‘qtl’ package, as described in Corty and Valdar (2019) <doi:10.1534/g3.118.200642>.

Depends R (>= 3.3.0)

License GPL-3

Encoding UTF-8

LazyData true

RoxygenNote 6.1.1

VignetteBuilder knitr

Imports doParallel, foreach, iterators, parallel, knitr, dplyr, dglm, evd, ggplot2, gtools, lazyeval, stringr, tidyr, testthat, purrr, qtl

Suggests covr

NeedsCompilation no

Author Robert Corty [aut, cre]

Maintainer Robert Corty <rcorty@gmail.com>

Repository CRAN

Date/Publication 2019-03-06 15:20:02 UTC

R topics documented:

c.scanonevar ................................................................. 2
effects_over_genome_plot .............................................. 2
is.scanonevar ............................................................... 3
mean_var_plot_model_free .............................................. 4
Description

combines scanonevar objects that have permutations to improve the precision of the p-value estimates.

Usage

```r
## S3 method for class 'scanonevar'
c(...)
```

Arguments

... the scanonevar objects with permutations to be combined

Value

a scanonevar object that is the concatenation of the inputted scanonevars

description

Plots estimated effects and their standard errors at each locus in the genome.

Usage

```r
effects_over_genome_plot(sov, covar_name_regex = ".", effect_type_regex = "(mean|var)", transform_var_effects = TRUE, se_ribbons = TRUE)
```
## is.scanonevar

### Arguments

- `sov`: the scanonevar
- `covar_name_regex`: regex that matches the covars we want to plot
- `effect_type_regex`: regex that matches 'mean', 'var', or both
- `transform_var_effects`: combine variance effects w intercept and exponentiate?
- `se_ribbons`: Should a ribbon from estimate - se to estimate + se be plotted?

### Value

- the plot

---

### Description

utilities for working with scanonevar objects

### Usage

- `is.scanonevar(x)`
- `is.scanonevar.w.perms(x)`
- `is.cross(x)`
- `is.f2.cross(x)`
- `is.f2.cross(x)`
- `is.cross.w.genoprobs(x)`

### Arguments

- `x`: object being tested

### Value

TRUE is X is a scanonevar object, FALSE otherwise.
TRUE if x is a scanone var with perms (typically, outputted from scanonevar.perm), and FALSE otherwise.
TRUE if x is a cross object, FALSE otherwise.
TRUE if x is a cross object of type F2, FALSE otherwise.
TRUE if x is a cross object of type 'bc' (backcross), FALSE otherwise
TRUE if x is a cross object with valid genoprobs for each chromosome, FALSE otherwise

Author(s)
Robert W. Corty <rcorty@gmail.com>
Robert W. Corty <rcorty@gmail.com>
Robert W. Corty <rcorty@gmail.com>
Robert W. Corty <rcorty@gmail.com>
Robert W. Corty <rcorty@gmail.com>

Examples

```r
is.scanonevar(x = 3)

test.cross <- qtl::sim.cross(map = qtl::sim.map(len = rep(20, 4), n.mar = 5))
test.cross <- qtl::calc.genoprob(cross = test.cross, step = 2)

x <- scanonevar(cross = test.cross)
is.scanonevar(x)

is.cross(3)
is.cross(qtl::sim.cross(map = qtl::sim.map()))

is.cross(3)
is.cross(qtl::sim.cross(map = qtl::sim.map()))

is.cross(3)
is.cross(qtl::sim.cross(map = qtl::sim.map()))

a <- qtl::sim.cross(map = qtl::sim.map())
is.cross.w.genoprobs(x = a)
b <- qtl::calc.genoprob(cross = a)
is.cross.w.genoprobs(x = b)
```

Description

plots with mean along the x axis and standard deviation along the y axis
plotting functions for package vqtl
mean_var_plot_model_free

Usage

mean_var_plot_model_free(cross, phenotype.name, grouping.factor.names, title = paste(phenotype.name, "by", paste(grouping.factor.names, collapse = ", ")))

mean_var_plot_model_based(cross, phenotype.name, focal.groups = NULL, nuisance.groups = NULL, genotype.names = c("AA", "AB", "BB"), xlim = NULL, ylim = NULL, title = paste(phenotype.name, "by", paste(focal.groups, collapse = ", ")), draw_ribbons = TRUE, se_line_size = 1, point_size = 1)

phenotype_at_marker_plot(cross, phenotype.name, marker_name, color_by = NULL, shape_by = NULL, point_alpha = 1, point_size = 1, Ibars = TRUE, connectIbars = TRUE, genotype_labels = NULL)

Arguments

cross	the cross
phenotype.name	the name of the phenotype of interest
grouping.factor.names	the factors by which the units are grouped
title	plot title
focal.groups	the focal covariates, whose effects will be plotted. Markers or phenotypes.
nuisance.groups	the nuisance covariates, whose effects will be modeled, then marginalized over. Markers or phenotypes.
genotype.names	plotting names of genotype groups
xlim	x axis limits
ylim	Y axis limits
draw_ribbons	Should ribbons be drawn connecting the sub-groups of the focal groups?
se_line_size	thickness of the lines indicating standard error
point_size	size of the plotted points
phenotype.name	The phenotype to plot
marker_name	The marker to stratify observations by
color_by	variable name to color the points by
shape_by	a discrete phenotype to map to the shape aesthetic of the points
point_alpha	alpha value (see-throughness) of the plotted points
Ibars	Should I bars be plotted showing the standard deviation of each group?
connectIbars	Should the Ibars be connected horizontally?
genotype_labels	plotting labels for genotype groups
plot.scanonevar

Value

Nothing, just plot.
nothing, just the plot.
nothing. Just plots.

Description

plot.scanonevar implements the plot generic for objects of class 'scanonevar'. Because scanonevar objects can be viewed in terms of LODs or empirical p-values, this plotting function checks the 'units' attribute to determine which to plot.

Usage

## S3 method for class 'scanonevar'
plot(x, y = NULL,
     chrs = unique(x[["result"]][["chr"]]), tests_to_plot = c("mQTL", "vQTL", "mvQTL"), plotting.units = if (any(grepl(pattern = "empir.p", x = names(x[["result"]])))) { "empir.p" } else { "LOD" },
     plot.title = x[["meta"]][["scan.formulae"]][["mean.alt.formula"]][[2]],
     marker.rug = TRUE, ymax = NULL, legend_pos = NULL,
     alpha_pos = c("left", "right", "none"), alpha_chr = 1,
     alpha_size = 2, ...)

Arguments

x the scanonevar object to be plotted
y Optionally, a scanone object to be plotting for comparison to the scanonevar object.
chrs Optionally, the subset of the chromosomes to plot
tests_to_plot which one or ones of the three possible tests to plot ('mQTL', 'vQTL', and 'mvQTL')
plotting.units One of 'LOD', 'asymp.p', or 'empir.p', implying whether LOD scores, asymptotic p-values, or empirical p-values should be plotted. Defaults to 'LOD'
plot.title the title of the plot
marker.rug Should a marker rug be plotted? Defaults to TRUE.
ymax the top of the y axis
legend_pos the position of the legend
alpha_pos the position of the alpha values (false positive rate)
alpha_chr which chromosome to put the alphas (FPRs) on
alpha_size size of annotations for alpha=0.05 and alpha=0.01 lines
... additional plotting arguments
**Details**

If such a strong signal was observed that the empirical p-value underflows R’s float type, this function produces an error. The author is open to suggestions on how to deal with this situation better.

These plots look better when both x (the scanonevar object) and y (optional scanone for comparison) are in units p values than when they are in LOD units.

**Value**

Returns the plot.

**Author(s)**

Robert Corty <rcorty@gmail.com>

**Examples**

```r
cross <- qtl::sim.cross(map = qtl::sim.map(len = rep(20, 3), n.mar = 5), n.ind = 50)
sim.sov <- scanonevar(cross = test.cross)
plot(x = test.sov)
```

---

**pve**

*percent variance explained*

**Description**

percent variance explained

**Usage**

`pve(LOD, n)`

**Arguments**

- **LOD**: the log odds between the null and alternative model
- **n**: the number of observations

**Value**

pve
scanonevar conducts a genome scan in an experimental cross, accommodating covariate effects in residual variance and identifying genetic effects on residual variance.

Usage

\[
\text{scanonevar}(\text{cross}, \text{mean.formula} = \text{phenotype} \sim \text{mean.QTL.add} + \text{mean.QTL.dom}, \\
\text{var.formula} = \sim \text{var.QTL.add} + \text{var.QTL.dom}, \text{chrs} = qtl::chrnames(\text{cross} = \text{cross}), \text{scan.types} = c(\text{"mQTL"}, \text{"vQTL"}, \text{"mvQTL"}), \\
\text{glm.family} = \text{"gaussian"}, \text{return.covar.effects} = \text{FALSE})
\]

Arguments

- \text{cross} \hspace{1cm} \text{The cross, built by qtl to be used in mapping}
- \text{mean.formula} \hspace{1cm} \text{The formula to describe the mean of the phenotype. Keywords are mean.QTL.add and mean.QTL.dom for the additive and dominance components of the QTL effect on the mean. dglm model will be fit if mean.formula has only fixed effects. hglm model will be fit if mean.formula has one or more random effects.}
- \text{var.formula} \hspace{1cm} \text{The formula to describe the residual variance of the phenotype. Keywords are var.QTL.add and var.QTL.dom for the additive and dominance components of the QTL effect on residual phenotype variance. var.formula must have only fixed effects.}
- \text{chrs} \hspace{1cm} \text{chromosomes to scan}
- \text{scan.types} \hspace{1cm} \text{a vector containing at least one of 'mQTL', 'vQTL', and 'mvQTL', or up to all three.}
- \text{glm.family} \hspace{1cm} \text{a character vector indicating the GLM family – either 'gaussian' or 'poisson'}
- \text{return.covar.effects} \hspace{1cm} \text{Should covariate effects estimated at each locus be returned?}

Value

27599

Author(s)

Robert W. Corty <rcorty@gmail.com>

Examples

\[
\text{set.seed}(27599) \\
\text{test.cross} \leftarrow \text{qtl::sim.cross}(\text{map} = \text{qtl::sim.map}(\text{len} = \text{rep}(20, 5), \text{n.mar} = 5), \text{n.ind} = 50) \\
\text{scanonevar}(\text{cross} = \text{test.cross})
\]
**Description**

`scanonevar.boot` conducts a nonparametric bootstrap of one chromosome to establish a confidence interval on any peaks.

**Usage**

```r
scanonevar.boot(sov, n.resamples, chr, qtl_type = c("mQTL", "vQTL", "mvQTL"), random.seed = 27599, n.cores = parallel::detectCores() - 2, silent = FALSE)
```

**Arguments**

- `sov`: the scanonevar whose significance should be assessed empirically in an FWER-controlling method.
- `n.resamples`: the number of resamples.
- `chr`: which chromosome to focus on.
- `qtl_type`: which type of QTL did you detect and want a CI for? mQTL, vQTL, or mvQTL.
- `random.seed`: value to start the random number generator at, for reproducibility.
- `n.cores`: number of cores to use for the permutations.
- `silent`: Should all messaging be suppressed?

**Value**

27599

**Author(s)**

Robert W. Corty <rcorty@gmail.com>

**Examples**

```r
set.seed(27599)
test.cross <- qtl::sim.cross(map = qtl::sim.map(len = rep(20, 5), n.mar = 5), n.ind = 50)
sov <- scanonevar(cross = test.cross)
```
scanonevar.perm conducts many permuted forms of the scanonevar inputted, to assess the statistical significance of the results in the inputted scanonevar in a FWER-controlling manner.

Usage

```
scanonevar.perm(sov, n.perms, random.seed = 27599, 
    n.cores = parallel::detectCores() - 2, silent = TRUE)
```

Arguments

- **sov**: the scanonevar whose significance should be assessed empirically in an FWER-controlling method
- **n.perms**: the number of permutations to do
- **random.seed**: value to start the random number generator at, for reproducibility
- **n.cores**: number of cores to use for the permutations
- **silent**: Should all messaging be suppressed?

Value

27599

Author(s)

Robert W. Corty <rcorty@gmail.com>

Examples

```
set.seed(27599)

test.cross <- qtl::sim.cross(map = qtl::sim.map(len = rep(20, 5), n.mar = 5), n.ind = 50)

scanonevar(cross = test.cross)
```
**summary.scanonevar**

**Description**

`summary.scanonevar` prints out the loci in a scanonevar object that exceed `thresh`. It is an S3 generic for `summary()`. It handles scanonevar objects in both LOD units and empirical p value units.

**Usage**

```r
## S3 method for class 'scanonevar'
summary(object, units = c("lod", "asymp.p", "empir.p"), thresh, ...)
```

**Arguments**

- `object` the scanonevar object to be summarized
- `units` Which units should be used to summarise? 'lod', 'asymp.p', or 'empir.p'
- `thresh` the threshold over which (for LODs) or under which (for empirical p values) a locus will be printed.
- `...` additional arguments controlling the summary

**Details**

none

**Value**

None. Only prints results to screen.

**Author(s)**

Robert Corty <rcorty@gmail.com>
Index

c.scanonevar, 2

effects_over_genome_plot, 2

is.cross(is.scanonevar), 3
is.f2.cross(is.scanonevar), 3
is.scanonevar, 3

mean_var_plot_model_based
(mean_var_plot_model_free), 4
mean_var_plot_model_free, 4

phenotype_at_marker_plot
(mean_var_plot_model_free), 4
plot.scanonevar, 6
pve, 7

scanonevar, 8
scanonevar.boot, 9
scanonevar.perm, 10
summary.scanonevar, 11