Package 'rehh'

May 9, 2026

Version 3.2.3

License GPL (>= 2)

Title Searching for Footprints of Selection using 'Extended Haplotype Homozygosity' Based Tests

Description Population genetic data such as 'Single Nucleotide Polymorphisms' (SNPs) is often used to identify genomic regions that have been under recent natural or artificial selection and might provide clues about the molecular mechanisms of adaptation. One approach, the concept of an 'Extended Haplotype Homozygosity' (EHH), introduced by (Sabeti 2002) <doi:10.1038/nature01140>, has given rise to several statistics designed for whole genome scans. The package provides functions to compute three of these, namely: 'iHS' (Voight 2006) <doi:10.1371/journal.pbio.0040072> for detecting positive or 'Darwinian' selection within a single population as well as 'Rsb' (Tang 2007) <doi:10.1371/journal.pbio.0050171> and 'XP-EHH' (Sabeti 2007) <doi:10.1038/nature06250>, targeted at differential selection between two populations. Various plotting functions are included to facilitate visualization and interpretation of these statistics.

Depends R (>= 2.10)

Imports methods, rehh.data

Suggests ape, bookdown, data.table, gap, knitr, qqman, rmarkdown, R.utils, testthat, vcfR

VignetteBuilder knitr

NeedsCompilation yes

Encoding UTF-8

RoxygenNote 7.3.3

URL https://CRAN.R-project.org/package=rehh, https://gitlab.com/oneoverx/rehh

BugReports https://gitlab.com/oneoverx/rehh/-/issues

Author Alexander Klassmann [aut, cre],
Mathieu Gautier [aut],
Renaud Vitalis [aut]

Maintainer Alexander Klassmann <rehh@oneoverx.eu>

Repository CRAN

Date/Publication 2026-01-30 13:00:02 UTC

Contents

rehh-package *rehh: Searching for Footprints of Selection using 'Extended Haplotype Homozygosity' Based Tests*

Description

Population genetic data such as 'Single Nucleotide Polymorphisms' (SNPs) is often used to identify genomic regions that have been under recent natural or artificial selection and might provide clues about the molecular mechanisms of adaptation. One approach, the concept of an 'Extended Haplotype Homozygosity' (EHH), introduced by (Sabeti 2002) doi:10.1038/nature01140, has given rise to several statistics designed for whole genome scans. The package provides functions to compute three of these, namely: 'iHS' (Voight 2006) doi:10.1371/journal.pbio.0040072 for detecting positive or 'Darwinian' selection within a single population as well as 'Rsb' (Tang 2007) doi:10.1371/journal.pbio.0050171 and 'XP-EHH' (Sabeti 2007) doi:10.1038/nature06250, targeted at differential selection between two populations. Various plotting functions are included to facilitate visualization and interpretation of these statistics.

Details

See `vignette("rehh", package = "rehh")` for an overview of the package and `vignette("examples", package = "rehh")` for a more detailed discussion of two small example data sets.

Author(s)

Maintainer: Alexander Klassmann <rehh@oneoverx.eu>

Authors:

- Mathieu Gautier <mathieu.gautier@inrae.fr>
- Renaud Vitalis

References

Gautier, M. and Naves, M. (2011). Footprints of selection in the ancestral admixture of a New World Creole cattle breed. *Molecular Ecology*, **20**, 3128-3143.

Gautier M. and Vitalis R. (2012). rehh: An R package to detect footprints of selection in genome-wide SNP data from haplotype structure. *Bioinformatics*, **28**(8), 1176-1177.

Gautier M., Klassmann A., and Vitalis R. (2017). rehh 2.0: a reimplementation of the R package rehh to detect positive selection from haplotype structure. *Molecular Ecology Resources*, **17**, 78-90.

Klassmann, A. and Gautier, M. (2022). Detecting selection using extended haplotype homozygosity (EHH)-based statistics in unphased or unpolarized data. *PLoS One*. **17**(1):e0262024 doi:10.1371/journal.pone.0262024

Sabeti, P.C. et al. (2002). Detecting recent positive selection in the human genome from haplotype structure. *Nature*, **419**, 832-837.

Sabeti, P.C. et al. (2007). Genome-wide detection and characterization of positive selection in human populations. *Nature*, **449**, 913-918.

Tang, K. and Thornton, K.R. and Stoneking, M. (2007). A New Approach for Using Genome Scans to Detect Recent Positive Selection in the Human Genome. *Plos Biology*, **7**, e171.

Voight, B.F. and Kudaravalli, S. and Wen, X. and Pritchard, J.K. (2006). A map of recent positive selection in the human genome. *Plos Biology*, **4**, e72.

See Also

Useful links:

- https://CRAN.R-project.org/package=rehh
- https://gitlab.com/oneoverx/rehh
- Report bugs at https://gitlab.com/oneoverx/rehh/-/issues

---

allelefurcation-class *An S4 class containing furcation trees for one allele of a focal marker*

Description

An S4 class containing the furcation trees for both sides of a focal marker for one allele.

Slots

`allele` the allele of the focal marker.

`description` "ancestral", "derived", "major", "minor", etc.

`count` the number of chromosomes with that allele.

`left` furcation tree to the left of the marker.

`right` furcation tree to the right of the marker.

See Also

`ftree`, `furcation`

---

as.newick *Convert a furcation tree into Newick format*

Description

Convert a furcation tree into Newick format.

Usage

```
as.newick(furcation, allele = 0, side, hap.names = seq_len(furcation@nhap))
```

Arguments

furcation	an object of furcation-class.
allele	the allele to be considered (default 0).
side	side (either "left" or "right").
hap.names	names/labels of chromosomes in haplotype data file. Per default haplotypes are numbered by their order in the input file.

See Also

ftree-class, calc_furcation, plot.furcation

Examples

```
#example haplohh object (280 haplotypes, 1424 SNPs)
#see ?haplohh_cgu_bta12 for details
data(haplohh_cgu_bta12)
#calculate furcation for the marker "F1205400"
#which displays a strong signal of selection
f <- calc_furcation(haplohh_cgu_bta12, mrk = "F1205400")
#get left tree of ancestral allele (coded as '0')
as.newick(f, 0, "left")
```

---

calc_candidate_regions

Determine candidate regions of selection

Description

Determine candidate regions of selection.

Usage

```
calc_candidate_regions(
  scan,
  threshold = NA,
  negativeThreshold = NA,
  pval = FALSE,
  ignore_sign = FALSE,
  window_size = 1e+06,
  overlap = 0,
  right = TRUE,
  min_n_mrk = 1,
  min_n_extr_mrk = 1,
  min_perc_extr_mrk = 0,
  join_neighbors = TRUE,
  keepNA = FALSE
)
```

Arguments

scan	a data frame containing scores (output of ihh2ihs, ines2rsb or ies2xpehh).
threshold	a positive numeric value. Scores which are higher are considered extreme.
negativeThreshold	a negative numeric value. Scores which are below are considered extreme.
pval	logical. If TRUE use the (negative log-) p-value instead of the score.
ignore_sign	logical. If TRUE, take absolute values of score.
window_size	size of sliding windows. If set to 1, no windows are constructed and only the individual extremal markers are reported.
overlap	size of window overlap (default 0, i.e. no overlap). Note that if you use this option together with join_neigbhors=TRUE, candidate regions might get bigger since the markers with extreme scores re-appear in several windows
right	logical, indicating if the windows should be closed on the right (and open on the left) or vice versa.
min_n_mrk	minimum number of markers per window.
min_n_extr_mrk	minimum number of markers with extreme value in a window.
min_perc_extr_mrk	minimum percentage of extremal markers among all markers.
join_neighbors	logical. If TRUE (default), merge neighboring windows with extreme values to a bigger interval.
keepNA	keep markers with a value of NA, i.e. for which no score could be calculated (e.g. due to a too small minor allele frequency). This option will affect the calculated number of markers in a window.

Details

There is no generally agreed method how to determine genomic regions which might have been under recent selection. Since selection tends to yield clusters of markers with outlier values, a common approach is to search for regions with an elevated number or fraction of outlier or extremal markers. This function allows to set three conditions a window must fulfill in order to classify as candidate region:

- min_n_mrk a minimum number of (any) markers.
- min_n_extr_mrk a minimum number of markers with outlier / extreme value.
- min_perc_extr_mrk a minimum percentage of extremal markers among all markers.

"Extreme" markers are defined by having a score above the specified threshold.

Value

A data frame with chromosomal regions, i.e. windows that fulfill the necessary conditions to qualify as candidate regions under selection. For each region the overall number of markers, their mean and maximum, the number of markers with extreme values, their percentage of all markers and their average are reported. In case only a positive threshold is specified, only positive scores are taken into account for the calculation of mean and maximum values. Vice versa for only a negative threshold being specified. In case both thresholds are specified, the absolute scores are used for mean and max.

See Also

calc_region_stats

Examples

```
#toy example of an ihs scan
scan <- data.frame(CHR = "1", POSITION = c(2, 3, 6, 7, 8) * 10000, IHS = c(-4, 0.5, 1, 6, NA))
scan
#candidate regions with default window size
calc_candidate_regions(scan, threshold = 2)
#with smaller window size
calc_candidate_regions(scan, threshold = 2, window_size = 20000)
#add negative threshold
calc_candidate_regions(scan, threshold = 2, negativeThreshold = -2, window_size = 20000)
#ignoring sign yields the same
calc_candidate_regions(scan, threshold = 2, ignore_sign = TRUE, window_size = 20000)
#'#use overlapping windows
calc_candidate_regions(scan, threshold = 2, ignore_sign = TRUE, window_size = 20000,
overlap = 10000)
#do not join windows with extreme values
calc_candidate_regions(scan, threshold = 2, ignore_sign = TRUE, window_size = 20000,
overlap = 10000, join_neighbors = FALSE)
#include windows without extreme values by 'min_n_extr_mrk = 0'
calc_candidate_regions(scan, threshold = 2, ignore_sign = TRUE, window_size = 20000,
overlap = 10000, join_neighbors = FALSE, min_n_extr_mrk = 0)
#include markers without score by 'keepNA = TRUE'
calc_candidate_regions(scan, threshold = 2, ignore_sign = TRUE, window_size = 20000,
overlap = 10000, join_neighbors = FALSE, min_n_extr_mrk = 0, keepNA = TRUE)
#include windows without markers by 'min_n_mrk = 0'
calc_candidate_regions(scan, threshold = 2, ignore_sign = TRUE, window_size = 20000,
overlap = 10000, join_neighbors = FALSE, min_n_mrk = 0, min_n_extr_mrk = 0, keepNA = TRUE)
```

---

calc_ehh *EHH and iHH computation for a given focal marker*

Description

Compute Extended Haplotype Homozygosity (EHH) and integrated EHH (iHH) for a given focal marker.

Usage

```
calc_ehh(
  haplohh,
  mrk,
  limhaplo = 2,
  limhomohaplo = 2,
  limehh = 0.05,
  include_zero_values = FALSE,
```

```
  include_nhaplo = FALSE,
  phased = TRUE,
  polarized = TRUE,
  scalegap = NA,
  maxgap = NA,
  discard_integration_at_border = TRUE,
  lower_y_bound = limehh,
  interpolate = TRUE
)
```

Arguments

haplohh	an object of class haplohh (see data2haplohh).
mrk	integer representing the number of the focal marker within the haplohh object or string representing its ID/name.
limhaplo	if there are less than limhaplo chromosomes that can be used for the calculation of EHH, the calculation is stopped. The option is intended for the case of missing data, which leads to the successive exclusion of haplotypes: the further away from the focal marker the less haplotypes contribute to EHH.
limhomohaplo	if there are less than limhomohaplo homozygous chromosomes, the calculation is stopped. This option is intended for unphased data and should be invoked only if relatively low frequency variants are not filtered subsequently (see main vignette and Klassmann et al. 2020).
limehh	limit at which EHH stops to be evaluated
include_zero_values	logical. If FALSE, return values only for those positions where the calculation is actually performed, i.e. until stopped by reaching either limehh or limhaplo. If TRUE, report EHH values for all markers, the additional ones being zero.
include_nhaplo	logical. If TRUE, report the number of evaluated haplotypes at each marker (only informative, if missing data leads to a decrease of evaluated haplotypes).
phased	logical. If TRUE (default) chromosomes are expected to be phased. If FALSE, the haplotype data is assumed to consist of pairwise ordered chromosomes belonging to diploid individuals. EHH is then estimated over individuals which are homozygous at the focal marker.
polarized	logical. TRUE by default. If FALSE, use major and minor allele instead of ancestral and derived. If there are more than two alleles then the minor allele refers to the second-most frequent allele.
scalegap	scale or cap gaps larger than the specified size to the specified size (default=NA, i.e. no scaling).
maxgap	maximum allowed gap in bp between two markers. If exceeded, further calculation of EHH is stopped at the gap (default=NA, i.e no limitation).
discard_integration_at_border	logical. If TRUE (default) and computation reaches first or last marker or a gap larger than maxgap, iHH is set to NA.
lower_y_bound	lower y boundary of the area to be integrated over (default: limehh). Can be set to zero for compatibility with the program hapbin.

`interpolate` logical. Affects only IHH values. If `TRUE` (default), integration is performed over a continuous EHH curve (values are interpolated linearly between consecutive markers), otherwise the EHH curve decreases stepwise at markers.

Details

Values for allele-specific Extended Haplotype Homozygosity (EHH) are computed upstream and downstream of the focal marker for each of its alleles. These values are integrated with respect to their genomic positions to yield an 'integrated EHH' (iHH) value for each allele.

Value

The returned value is a list containing the following elements:

mrk.name The name/identifier of the focal marker.

freq A vector with the frequencies of the alleles of the focal marker.

ehh A data frame with EHH values for each allele of the focal marker.

ihh A vector with iHH (integrated EHH) values for each allele of the focal marker.

References

Gautier, M. and Naves, M. (2011). Footprints of selection in the ancestral admixture of a New World Creole cattle breed. *Molecular Ecology*, **20**, 3128-3143.

Klassmann, A. and Gautier, M. (2022). Detecting selection using extended haplotype homozygosity (EHH)-based statistics in unphased or unpolarized data. *PLoS One*. **17**(1):e0262024 doi:10.1371/journal.pone.0262024

Sabeti, P.C. et al. (2002). Detecting recent positive selection in the human genome from haplotype structure. *Nature*, **419**, 832-837.

Sabeti, P.C. et al. (2007). Genome-wide detection and characterization of positive selection in human populations. *Nature*, **449**, 913-918.

Tang, K. and Thornton, K.R. and Stoneking, M. (2007). A New Approach for Using Genome Scans to Detect Recent Positive Selection in the Human Genome. *Plos Biology*, **7**, e171.

Voight, B.F. and Kudaravalli, S. and Wen, X. and Pritchard, J.K. (2006). A map of recent positive selection in the human genome. *Plos Biology*, **4**, e72.

See Also

`data2haplohh`, `plot.ehh`, `calc_ehhs`, `scan_hh`.

Examples

```
#example haplohh object (280 haplotypes, 1424 SNPs)
#see ?haplohh_cgu_bta12 for details
data(haplohh_cgu_bta12)
#computing EHH statistics for the marker "F1205400"
#which displays a strong signal of selection
ehh <- calc_ehh(haplohh_cgu_bta12, mrk = "F1205400")
```

calc_ehhs *EHHS and iES computation for a given focal marker*

Description

Compute site-specific Extended Haplotype Homozygosity (EHHS) and integrated EHHS (iES) for a given focal marker.

Usage

```
calc_ehhs(
  haplohh,
  mrk,
  limhaplo = 2,
  limhomohaplo = 2,
  limehhs = 0.05,
  include_zero_values = FALSE,
  include_nhaplo = FALSE,
  phased = TRUE,
  scalegap = NA,
  maxgap = NA,
  discard_integration_at_border = TRUE,
  lower_y_bound = limehhs,
  interpolate = TRUE
)
```

Arguments

haplohh	an object of class haplohh (see data2haplohh).
mrk	integer representing the number of the focal marker within the haplohh object or string representing its ID/name.
limhaplo	if there are less than limhaplo chromosomes that can be used for the calculation of EHH, the calculation is stopped. The option is intended for the case of missing data, which leads to the successive exclusion of haplotypes: the further away from the focal marker the less haplotypes contribute to EHH.
limhomohaplo	if there are less than limhomohaplo homozygous chromosomes, the calculation is stopped. This option is intended for unphased data and should be invoked only if relatively low frequency variants are not filtered subsequently (see main vignette and Klassmann et al. 2020).
limehhs	limit at which EHHS stops to be evaluated.
include_zero_values	logical. If FALSE, return values only for those positions where the calculation is actually performed, i.e. until stopped by reaching either limehh or limhaplo. If TRUE, report EHH values for all markers, the additional ones being zero.
include_nhaplo	logical. If TRUE, report the number of evaluated haplotypes at each marker (only informative, if missing data leads to a decrease of evaluated haplotypes).

`phased` logical. If `TRUE` (default) chromosomes are expected to be phased. If `FALSE`, the haplotype data is assumed to consist of pairwise ordered chromosomes belonging to diploid individuals. EHHS is then estimated over individuals which are homozygous at the focal marker.

`scalegap` scale or cap gaps larger than the specified size to the specified size (default=`NA`, i.e. no scaling).

`maxgap` maximum allowed gap in bp between two markers. If exceeded, further calculation of EHHS is stopped at the gap (default=`NA`, i.e no limitation).

`discard_integration_at_border` logical. If `TRUE` (default) and computation reaches first or last marker or a gap larger than `maxgap`, iHH is set to `NA`.

`lower_y_bound` lower y boundary of the area to be integrated over (default: `limehhs`). Can be set to zero for compatibility with the program hapbin.

`interpolate` logical. Affects only IES and INES values. If `TRUE` (default), integration is performed over a continuous EHHS curve (values are interpolated linearly between consecutive markers), otherwise the EHHS curve decreases stepwise at markers.

Details

Values for site-specific Extended Haplotype Homozygosity (EHHS) are computed at each position upstream and downstream of the focal marker. These values are integrated with respect to their genomic position to yield an 'integrated EHHS' (iES) value.

Value

The returned value is a list containing the following elements:

mrk.name The name/identifier of the focal marker.

ehhs A table containing EHHS values as used by Sabeti et al. (2007), resp. the same values normalized to 1 at the focal marker (nEHHS) as used by Tang et al. (2007).

IES Integrated EHHS.

INES Integrated normalized EHHS.

References

Gautier, M. and Naves, M. (2011). Footprints of selection in the ancestral admixture of a New World Creole cattle breed. *Molecular Ecology*, **20**, 3128-3143.

Klassmann, A. and Gautier, M. (2022). Detecting selection using extended haplotype homozygosity (EHH)-based statistics in unphased or unpolarized data. *PLoS One*. **17**(1):e0262024 doi:10.1371/journal.pone.0262024

Sabeti, P.C. et al. (2002). Detecting recent positive selection in the human genome from haplotype structure. *Nature*, **419**, 832-837.

Sabeti, P.C. et al. (2007). Genome-wide detection and characterization of positive selection in human populations. *Nature*, **449**, 913-918.

Tang, K. and Thornton, K.R. and Stoneking, M. (2007). A New Approach for Using Genome Scans to Detect Recent Positive Selection in the Human Genome. *Plos Biology*, **7**, e171.

Voight, B.F. and Kudaravalli, S. and Wen, X. and Pritchard, J.K. (2006). A map of recent positive selection in the human genome. *Plos Biology*, **4**, e72.

See Also

data2haplohh, plot.ehhs, calc_ehh, scan_hh.

Examples

```
#example haplohh object (280 haplotypes, 1424 SNPs)
#see ?haplohh_cgu_bta12 for details
data(haplohh_cgu_bta12)
#computing EHHS statistics for the marker "F1205400"
#which displays a strong signal of selection
ehhs <- calc_ehhs(haplohh_cgu_bta12, mrk = "F1205400")
```

---

calc_furcation *calculate furcation trees around a focal marker*

Description

Calculate furcation trees around a focal marker. A furcation tree captures in greater detail than EHH values the decrease of extended haplotype homozygosity at increasing distances from the selected focal marker.

Usage

```
calc_furcation(
  haplohh,
  mrk,
  allele = NA,
  limhaplo = 2,
  phased = TRUE,
  polarized = TRUE
)
```

Arguments

haplohh	an object of class haplohh (see data2haplohh).
mrk	integer representing the number of the focal marker within the haplohh object or string representing its ID/name.
allele	a vector of alleles as coded internally, i.e. in case of polarized alleles, 0 represents the ancestral, 1 or higher the derived alleles. If NULL, all alleles of the focal marker are considered.
limhaplo	if there are less than limhaplo chromosomes that can be used for the calculation, it is stopped. This is useful in case of missing data, which lead to a successive exclusion of haplotypes: the further away from the focal marker the less haplotypes are evaluated.

phased	logical. If TRUE (default), chromosomes are expected to be phased. If FALSE, consecutive chromosomes are assumed to belong to diploid individuals and furcation trees are limited to within individuals which are homozygous at the focal marker.
polarized	logical. Affects only the order of furcations. If TRUE (default), the ancestral allele becomes the first furcation and derived alleles are sorted by their internal coding. Otherwise all alleles are sorted by their internal coding.

Details

A haplotype furcation tree visualizes the breakdown of LD at increasing distances from the focal marker. The root of each tree is an allele of the focal marker, which in turn is identified by a vertical dashed line. Moving either to the "left" or to the "right" of the focal marker, each further marker is an opportunity for a node; the tree either divides or does not, based on whether alleles at that marker distinguish between hitherto identical extended haplotypes. The thickness of the lines corresponds to the number of chromosomes sharing an extended haplotype.

Value

An object of class furcation, containing the furcation structure of the specified alleles at the focal marker.

References

Sabeti, P.C. and Reich, D.E. and Higgins, J.M. and Levine, H.Z.P and Richter, D.J. and Schaffner, S.F. and Gabriel, S.B. and Platko, J.V. and Patterson, N.J. and McDonald, G.J. and Ackerman, H.C. and Campbell, S.J. and Altshuler, D. and Cooper, R. and Kwiatkowski, D. and Ward, R. and Lander, E.S. (2002). Detecting recent positive selection in the human genome from haplotype structure. Nature, 419, 832-837.

See Also

plot.furcation, calc_haplen.

Examples

```
#example haplohh object (280 haplotypes, 1424 SNPs)
#see ?haplohh_cgu_bta12 for details
data(haplohh_cgu_bta12)
#plotting a furcation diagram for both ancestral and derived allele
#from the marker "F1205400"
#which display a strong signal of selection
f <- calc_furcation(haplohh_cgu_bta12, mrk = "F1205400")
plot(f)
```

calc_haplen *Calculate length of longest shared haplotypes around a focal marker*

Description

Calculate for each chromosome the maximum length of its extended haplotype homozygosity.

Usage

```
calc_haplen(furcation)
```

Arguments

furcation	an object of class furcation calculated by calc_furcation.

Details

Extended haplotype homozygosity is defined as the region around a focal marker in which a particular chromosome shares a haplotype with (its sequence is identical to) another chromosome. The function calculates for each chromosome the boundaries of its longest shared haplotype. These correspond to the last furcations of a chromsome in a furcation diagram. Note that the calculation is performed independently upstream and downstream of the focal marker and hence upper and lower boundaries do not necessarily arise from the same chromosomal pair.

Value

The functions returns a list containing four elements:

mrk.name name/identifier of the focal marker.

position position of the focal marker.

xlim positions of left- and rightmost markers covered by extended haplotypes.

haplen a data frame with the coordinates of extended haplotypes around the focal marker.

Examples

```
#example haplohh object (280 haplotypes, 1424 SNPs)
#see ?haplohh_cgu_bta12 for details
data(haplohh_cgu_bta12)
#plotting haplotype lengths for both ancestral and derived allele
#of the marker "F1205400"
#which displays a strong signal of selection
f <- calc_furcation(haplohh_cgu_bta12, mrk = "F1205400")
h <- calc_haplen(f)
plot(h)
```

`calc_pairwise_haplen` *Calculate pairwise shared haplotype length between all chromosomes*

Description

Calculate pairwise shared haplotype length between all chromosomes at a focal marker.

Usage

```
calc_pairwise_haplen(
  haplohh,
  mrk,
  phased = TRUE,
  maxgap = NA,
  max_extend = NA,
  side = "both"
)
```

Arguments

`haplohh`	an object of class `haplohh` (see `data2haplohh`).
`mrk`	integer representing the number of the focal marker within the haplohh object or string representing its ID/name.
`phased`	logical. If `TRUE` (default) chromosomes are expected to be phased. If `FALSE`, the haplotype data is assumed to consist of pairwise ordered chromosomes belonging to diploid individuals and only the two chromosomes of each individual are compared.
`maxgap`	maximum allowed gap in bp between two markers. If exceeded, further calculation is stopped at the gap (default=`NA`, i.e. no limitation).
`max_extend`	maximum distance in bp to extend shared haplotypes away from the focal marker. (default `NA`, i.e. no limitation).
`side`	side to consider, either "left" (positions lower than focal position), "right" (positions higher than focal position) or "both" (default).

Details

The function computes the length of shared haplotypes (stretches of identical sequence) around the focal marker.

Note that the function `calc_haplen` calculates for each chromosome the boundaries of its longest shared haplotype; separately upstream and downstream of the focal marker.

Value

The returned value is a matrix with pairwise shared haplotype lengths.

See Also

calc_haplen, data2haplohh, scan_hh_full.

Examples

```
#example haplohh object (280 haplotypes, 1424 SNPs)
#see ?haplohh_cgu_bta12 for details
data(haplohh_cgu_bta12)
#computing shared haplotype lengths around the marker "F1205400"
#which displays a strong signal of selection
m <- calc_pairwise_haplen(haplohh_cgu_bta12, mrk = "F1205400")
#note that calc_haplen() returns the length of the longest shared haplotype:
f <- calc_furcation(haplohh_cgu_bta12, mrk = "F1205400")
h <- calc_haplen(f)
#all pairwise shared haplotype lengths to the left of the focal marker
m_left <- calc_pairwise_haplen(haplohh_cgu_bta12, mrk = "F1205400", side = "left")
#get for each chromosome the maximum shared haplotype length
max_left <- apply(m_left, 1, max)
all.equal(max_left, h$position - h$haplen$MIN, check.attributes = FALSE)
#the same for shared haplotypes to the right of the focal marker
m_right <- calc_pairwise_haplen(haplohh_cgu_bta12, mrk = "F1205400", side = "right")
max_right <- apply(m_right, 1, max)
all.equal(max_right, h$haplen$MAX - h$position, check.attributes = FALSE)
```

---

calc_region_stats *Calculate score statistics for given regions*

Description

Calculate score statistics (extremal values) for given regions. This function is intended for the comparison of different scores for the same chromosomal regions.

Usage

```
calc_region_stats(
  scan,
  regions,
  threshold = NA,
  pval = FALSE,
  ignore_sign = FALSE,
  right = TRUE
)
```

Arguments

scan	a data frame containing scores (output of ihh2ihs, ines2rsb or ies2xpehh).
regions	a data frame with column names CHR, START and END, specifying chromosomal regions (e.g. as obtained by function calc_candidate_regions).

threshold	boundary score above which markers are defined as "extreme".
pval	logical. If TRUE use the (negative log-) p-value instead of the score.
ignore_sign	logical. If TRUE (default), take absolute values of score.
right	logical, indicating if the regions should be closed on the right (and open on the left) or vice versa.

Value

A data frame with chromosomal regions. For each region the overall number of markers, their mean and maximum, the number of markers with extremal values, their percentage of all markers and their average are reported.

See Also

calc_candidate_regions

---

calc_sfs_tests *Calculate site frequency spectrum test statistics*

Description

Calculate site frequency spectrum (SFS) tests Tajima's D, Fay & Wu's H and Zeng's E.

Usage

```
calc_sfs_tests(
  haplohh,
  polarized = TRUE,
  window_size = NA,
  overlap = 0,
  right = TRUE,
  min_n_mrk = 1,
  verbose = TRUE
)
```

Arguments

haplohh	an object of class haplohh (see data2haplohh)
polarized	logical. TRUE by default. If FALSE, use major and minor allele instead of ancestral and derived. If there are more than two alleles then the minor allele refers to the second-most frequent allele. Note that Tajima's D remains unchanged, Fay & Wu's H is always zero for folded spectra and Zeng's E becomes equal to Tajima's D.
window_size	size of sliding windows. If NA (default), there will be only one window covering the whole length of the chromosome.
overlap	size of window overlap (default 0, i.e. no overlap).

`right`	logical, indicating if the windows should be closed on the right and open on the left (default) or vice versa.
`min_n_mrk`	minimum number of (polymorphic) markers per window.
`verbose`	logical. `TRUE` by default; reports if multi-allelic sites are removed.

Details

Neutrality tests based on the site frequency spectrum (SFS) are largely unrelated to EHH-based methods. The tests provided here are implemented elsewhere, too (e.g. in package PopGenome).

Each test compares two estimations of the *scaled mutation rate* theta, which all have the same expected value under neutrality. Deviations from zero indicate violations of the neutral null model, typically population size changes, population subdivision or selection. Tajima's D and Fay & Wu's H become negative in presence of an almost completed sweep, Zeng's E becomes positive for some time after it. Significance can typically be assigned only by simulations.

The standard definition of the tests cannot cope with missing values and typically markers with missing genotypes must be discarded. Ferretti (2012) provides an extension that can handle missing values (without discarding any non-missing values). In this package, only the first moments (the theta-estimators themselves) are adapted accordingly, but not the second moments (their variances), because the latter is computationally demanding and the resulting bias relatively small. It is recommended, though, to discard markers or haplotypes with more than 20% missing values.

Multi-allelic markers are always removed since the tests rely on the "infinite sites model" which implies that all polymorphic markers are bi-allelic. Monomorphic markers can be present, but are irrelevant for the tests.

Value

A data frame with window coordinates, the number of contained (polymorphic) markers, Watterson's, Tajima's and Zeng's estimators of theta and the test statistics of Tajima's D, Fay & Wu's H and Zeng's E.

References

Watterson, G.A. (1975). On the number of segregating sites in genetical models without recombination. *Theoretical Population Biology* **7**(2) 256-276.

Tajima, F. (1983). Evolutionary relationship of DNA sequences in finite populations. *Genetics* **105**(2) 437-60.

Tajima, F. (1989). Statistical method for testing the neutral mutation hypothesis by DNA polymorphism. *Genetics* **123**(3) 585-95.

Fay, J. and Wu, C. (2000). Hitchhiking under positive Darwinian selection. *Genetics* **155**(3) 1405-13.

Zeng, E. et al. (2006). Statistical tests for detecting positive selection by utilizing high-frequency variants. *Genetics* **174**(3) 1431-9.

Ferretti, L. and Raineri, E. and Ramos-Onsins, S. (2012). Neutrality tests for sequences with missing data. *Genetics* **191**(4) 1397-401.

Examples

```
make.example.files()
# neutral evolution
hh <- data2haplohh("example_neutral.vcf", verbose = FALSE)
calc_sfs_tests(hh)
# strong selective sweep
hh <- data2haplohh("example_sweep.vcf", verbose = FALSE)
calc_sfs_tests(hh)
remove.example.files()
```

`data2haplohh` *Convert data from input file to an object of class haplohh*

Description

Convert input data files to an object of `haplohh-class`.

Usage

```
data2haplohh(
  hap_file,
  map_file = NA,
  min_perc_geno.hap = NA,
  min_perc_geno.mrk = 100,
  min_maf = NA,
  chr.name = NA,
  popsel = NA,
  recode.allele = FALSE,
  allele_coding = "12",
  haplotype.in.columns = FALSE,
  remove_multiple_markers = FALSE,
  polarize_vcf = TRUE,
  capitalize_AA = TRUE,
  vcf_reader = "data.table",
  position_scaling_factor = NA,
  verbose = TRUE
)
```

Arguments

`hap_file`	file containing haplotype data (see details below).
`map_file`	file containing map information (see details below).
`min_perc_geno.hap`	threshold on percentage of missing data for haplotypes (haplotypes with less than `min_perc_geno.hap` percent of markers genotyped are discarded). Default is `NA`, hence no constraint.

`min_perc_geno.mrk`
: threshold on percentage of missing data for markers (markers genotyped on less than `min_perc_geno.mrk` percent of haplotypes are discarded). By default, `min_perc_geno.mrk=100`, hence only fully genotyped markers are retained. This value cannot be set to `NA` or zero.

`min_maf`
: threshold on the Minor Allele Frequency. Markers having a MAF lower than or equal to minmaf are discarded. In case of multi-allelic markers the second-most frequent allele is referred to as minor allele. Setting this value to zero eliminates monomorphic sites. Default is `NA`, hence no constraint.

`chr.name`
: name of the chromosome considered (relevant if data for several chromosomes is contained in the haplotype or map file).

`popsel`
: code of the population considered (relevant for fastPHASE output which can contain haplotypes from various populations).

`recode.allele`
: *Deprecated*. logical. `FALSE` by default. `TRUE` forces parameter `allele_coding` to `"map"`, `FALSE` leaves it unchanged.

`allele_coding`
: the allele coding provided by the user. Either `"12"` (default), `"01"`, `"map"` or `"none"`. The option is irrelevant for vcf files and ms output.

`haplotype.in.columns`
: logical. If `TRUE`, phased input haplotypes are assumed to be in columns (as produced by the SHAPEIT2 program (O'Connell et al., 2014).

`remove_multiple_markers`
: logical. If `FALSE` (default), conversion stops, if multiple markers with the same chromosomal position are encountered. If `TRUE`, duplicated markers are removed (all but the first marker with identical positions).

`polarize_vcf`
: logical. Only of relevance for vcf files. If `TRUE` (default), tries to polarize variants with help of the AA entry in the INFO field. Unpolarized alleles are discarded. If `FALSE`, allele coding of vcf file is used unchanged as internal coding.

`capitalize_AA`
: logical. Only of relevance for vcf files with ancestral allele information. Low confidence ancestral alleles are usually coded by lower-case letters. If `TRUE` (default), these are changed to upper case before the alleles of the sample are matched for polarization.

`vcf_reader`
: library used to read vcf. By default, low-level parsing is performed using the generic package `data.table`. In order to read compressed files, the package `R.utils` must be installed, too. If the specialized package `vcfR` is available, set this parameter to `"vcfR"`.

`position_scaling_factor`
: intended primarily for output of ms where positions lie in the interval [0,1]. These can be rescaled to sizes of typical markers in real data.

`verbose`
: logical. If `TRUE` (default), report verbose progress.

Details

Five haplotype input formats are supported:

- a "standard format" with haplotypes in rows and markers in columns (with no header, but a haplotype ID/name in the first column).

- a "transposed format" similar to the one produced by the phasing program SHAPEIT2 (O'Connell et al., 2014) in which haplotypes are in columns and markers in rows (with neither header nor marker IDs nor haplotype IDs).
- output files from the fastPHASE program (Sheet and Stephens, 2006). If haplotypes from several different population were phased simultaneously (-u fastPHASE option was used), it is necessary to specify the population of interest by parameter `popsel` (if this parameter is not or wrongly set, the error message will provide a list of the population numbers contained in the file).
- files in variant call format (vcf). No mapfile is needed is this case. If the file contains several chromosomes, it is necessary to choose one by parameter `chr.name`.
- output of the simulation program 'ms'. No mapfile is needed in this case. If the file contains several 'runs', a specific number has to be specified by the parameter `chr.name`.

The "transposed format" has to be explicitly set while the other formats are recognized automatically.

The map file contains marker information in three, or, if it is used for polarization (see below), five columns:

- marker name/id
- chromosome
- position (physical or genetic)
- ancestral allele encoding
- derived allele encoding

The markers must be in the same order as in the haplotype file. If several chromosomes are represented in the map file, it is necessary to choose that which corresponds to the haplotype file by parameter `chr.name`.

Haplotypes can be given either with alleles already coded as numbers (in two possible ways) or with the actual alleles (e.g. nucleotides) which can be translated into numbers either using the fourth and fifth column of the map file or by their alpha-numeric order. Correspondingly, the parameter `allele_coding` has to be set to either `"12"`, `"01"`, `"map"` or `"none"`:

- `"12"`: 0 represents missing values, 1 the ancestral allele and 2 (or higher integers) derived allele(s).
- `"01"`: `NA` or '.' (a point) represent missing values, 0 the ancestral and 1 (or higher integers) derived allele(s).
- `"map"`: for each marker, the fourth column of the map file defines the ancestral allele and the fifth column derived alleles. In case of multiple derived alleles, they must be separated by commas without space. Alleles in the haplotype file which do not appear in neither of the two columns of the map file are regarded as missing values (`NA`).
- `"none"`: `NA` or '.' (a point) represent missing values, otherwise for each marker the allele that comes first in alpha-numeric order is coded by 0, the next by 1, etc. Evidently, this coding does not convey any information about allele status as ancestral or derived, hence the alleles cannot be regarded as polarized.

The information of allelic ancestry is exploited only in the frequency-bin-wise standardization of iHS (see ihh2ihs). However, although ancestry status does not figure in the formulas of the cross populations statistics Rsb and XP-EHH, their values do depend on the assigned status.

The arguments min_perc_geno.hap, min_perc_geno.mrk and min_maf are evaluated in this order.

Value

The returned value is an object of haplohh-class.

References

Scheet P, Stephens M (2006) A fast and flexible statistical model for large-scale population genotype data: applications to inferring missing genotypes and haplotypic phase. *Am J Hum Genet*, **78**, 629-644.

O'Connell J, Gurdasani D, Delaneau O, et al (2014) A general approach for haplotype phasing across the full spectrum of relatedness. *PLoS Genet*, **10**, e1004234.

Examples

```
#copy example files into the current working directory.
make.example.files()
#create object using a haplotype file in "standard format"
hap <- data2haplohh(hap_file = "bta12_cgu.hap",
                    map_file = "map.inp",
                    chr.name = 12,
                    allele_coding = "map")
#create object using fastPHASE output
hap <- data2haplohh(hap_file = "bta12_hapguess_switch.out",
                    map_file = "map.inp",
                    chr.name = 12,
                    popsel = 7,
                    allele_coding = "map")
#clean up demo files
remove.example.files()
```

---

distribplot *Plot distribution of standardized iHS, Rsb or XP-EHH values*

Description

Plot the observed distribution of standardized iHS, Rsb or XP-EHH values together with the standard Gaussian distribution.

Usage

```
distribplot(
  data,
  lty = 1,
  lwd = 1.5,
  col = c("blue", "red"),
  qqplot = FALSE,
  resolution = 0.01,
  ...
)
```

Arguments

data	a vector of iHS, Rsb or XPEHH values.
lty	line type.
lwd	line width.
col	a vector describing the colors of the observed and Gaussian distribution, respectively.
qqplot	logical. If TRUE a qq-plot is drawn instead of the distribution density curve.
resolution	affects only qqplot. Rasterize data points to a quadratic grid with the specified resolution and remove duplicate points. Defaults to 0.01.
...	further arguments passed to plot.default.

Value

The function returns a plot.

See Also

ihh2ihs, ines2rsb, ies2xpehh, manhattanplot.

Examples

```
library(rehh.data)
#results from a genome scan (44,057 SNPs) see ?wgscan.cgu for details
data(wgscan.cgu)
#extract vector with iHS values from data frame
IHS <- ihh2ihs(wgscan.cgu)$ihs[["IHS"]]
distribplot(IHS, main = "iHS (CGU population)")
distribplot(IHS, main = "iHS (CGU population)", qqplot = TRUE)
```

extract_regions *Extract regions from a scan*

Description

Extract regions from a scan data frame.

Usage

```
extract_regions(scan, regions, right = TRUE)
```

Arguments

scan	A data frame with chromosomal positions like obtained by scan_hh, ihh2ihs, ines2rsb or ies2xpehh.
regions	A data frame with genomic regions like the output of calc_candidate_regions.
right	logical, indicating if the intervals should be closed on the right (and open on the left) or vice versa.

Value

A subset of data frame scan, retaining only positions belonging to the regions specified in data frame regions.

Examples

```
library(rehh.data)
data(wgscan.cgu)
regions <- data.frame(CHR = 12, START = 2.88e+7, END = 2.92e+7)
extract_regions(wgscan.cgu, regions)
```

freqbinplot *Plot of unstandardized iHS within frequency bins*

Description

Plot of unstandardized iHS within frequency bins.

Usage

```
freqbinplot(
  x,
  spectrum = FALSE,
  main = NA,
  xlab = "Derived allele frequency",
  ylab = NA,
  xlim = c(0, 1),
  ylim = NULL,
  pch = 20,
  ...
)
```

Arguments

x	data (output of function ihh2ihs)
spectrum	logical. If TRUE, plot frequency spectrum instead of iHS.
main	an overall title for the plot.
xlab	a title for the x axis.
ylab	a title for the y axis.
xlim	the x coordinate range of the plot.
ylim	the y coordinate range of the plot.
pch	plotting 'character' see points.
...	further arguments to be passed to plot resp. points.

Details

The plot shows the mean and the quantiles calculated by function ihh2ihs for the unstandardized iHS in each frequency bin. Note that the standardization of iHS is performed bin-wise in order to reduce the frequency-dependence of iHS values (expected under neutrality). An implicit assumption of this procedure is that each bin is dominated by neutral markers.

See Also

ihh2ihs

Examples

```
library(rehh.data)
data(wgscan.cgu)
#results from a genome scan (44,057 SNPs)
#see ?wgscan.eut and ?wgscan.cgu for details
wgscan.cgu.ihs <- ihh2ihs(wgscan.cgu)
freqbinplot(wgscan.cgu.ihs)
```

ftree-class *An S4 class to represent a furcation tree on one side of one allele of a focal marker*

Description

An S4 class to represent a furcation tree on one side of one allele of a focal marker

Details

A furcation structure consists of two trees ("left" and "right") for each allele of a focal marker. If there are only bi-allelic markers and no missing values, the trees are bifurcating.

Missing values are treated similarly to an extra allele in so far as they cause a furcation. However, the resulting daughter node is marked accordingly and the chromosomes excluded from further calculations. If all chromosomes of a parent node have missing values, the "furcation" is degenerated and yields a single daughter node.

Note that a tree with n leaves can have at most 2n-1 nodes.

In a furcation tree, the leaves do not necessarily represent single chromosomes, either due to multiple missing data or because the first/last marker was reached before all extended haplotypes were distinct.

Slots

`node_parent` a vector, representing the tree structure. Each node (number) is assigned its parent node (number).

`node_pos` a vector, assigning to each node (number) its position in the chromosome, i.e. at which marker position the furcation occurred.

`node_with_missing_data` a vector of type `logical`. Pseudo-furcations arise due to missing data at a marker. The daughter node (number) is marked accordingly.

`label_parent` a vector, that attaches an "extra leave", representing the haplotype number (defined by the order in the haplotype data file) to leaves of the tree. This is necessary because in general not all leaves of the original tree represent a single haplotype/chromosome.

furcation-class *An S4 class representing the complete furcation pattern around a focal marker.*

Description

An S4 class representing the complete furcation pattern around a focal marker.

Slots

`.Data` a list containing for each allele an object of `allelefurcation-class`.

`mrk.name` the name/identifier of the focal marker.

`position` the chromosomal position of the focal marker.

`xlim` the range of marker positions.

`nhap` the number of haplotypes in the sample.

See Also

`calc_furcation`

Examples

```
# copy example files into working directory
make.example.files()
# read first example file
hh <- data2haplohh(hap_file = "example1.hap", map_file = "example1.map", allele_coding = "01")
# remove example files
remove.example.files()
# calculate furcation structure around marker "rs6"
f <- calc_furcation(hh, mrk = "rs6")
# extract left side tree of ancestral allele (which is coded by '0')
f[['0']]@left
# the tree consists of seven nodes, '1' being the root node
# nodes 2 and 3 have the root node as parent, etc.
# the first chromosome is attached as a label node to node 7, etc.
# For comparison, a plot of the complete furcation structure:
plot(f)
```

---

haplen-class *class for haplotype length*

Description

class for haplotype length

haplohh-class *Class "haplohh"*

Description

An object of this class contains the information needed for computation of EHH based statistics.

Usage

```
## S4 method for signature 'haplohh'
chr.name(x)

## S4 method for signature 'haplohh'
positions(x)

## S4 method for signature 'haplohh'
haplo(x)

## S4 method for signature 'haplohh'
nmrk(x)

## S4 method for signature 'haplohh'
mrk.names(x)

## S4 method for signature 'haplohh'
nhap(x)

## S4 method for signature 'haplohh'
hap.names(x)
```

Arguments

x	an object of this class.

Details

This class is the basis for all calculations done by this package. Note that the matrix in slot haplo has to be of type integer, not numeric. Objects built by versions of rehh up to 2.0.4 coded this matrix as numeric and used a different coding scheme. They can be converted e.g. by haplohh <- update_haplohh(old_haplohh) in order be used with the present version.

Slots

chr.name name of the chromosome/scaffold to which the markers belong.

positions vector of type numeric containing the marker positions within the chromosome.

haplo matrix of type integer containing haplotypes in rows and markers in columns.

See Also

data2haplohh, update_haplohh

Examples

```
showClass("haplohh")
```

---

haplohh2sweepfinder *Translate object of* haplohh-class *into SweepFinder format*

Description

Extract allele frequencies of an object of class haplohh-class and returns a table in SweepFinder input format.

Usage

```
haplohh2sweepfinder(haplohh, polarized = TRUE, verbose = TRUE)
```

Arguments

haplohh	object of class haplohh-class.
polarized	logical. If TRUE (default), flag "folded" is set to 0, otherwise to 1.
verbose	logical. If TRUE (default), prints filter statements.

Details

SweepFinder and SweeD are two stand-alone programs which implement the same method to detect selective sweeps using the allele frequency at each site. This function calculates these frequencies from a haplohh-class and returns a table which can be saved into a file (with tabs as separators, without row names and quotes) that can be used as input for the two programs.

Sites with less than two haplotypes genotyped or with more than two alleles are removed. If polarized, sites monomorphic for the ancestral allele are removed, too.

Value

A dataframe with four columns:

- **position** marker position
- **x** (absolute) frequency of the alternative (derived) variant
- **n** number of non-missing genotypes
- **folded** a flag marking polarization

References

DeGiorgio, M., and, Huber, CD and Hubisz, MJ and, Hellmann, I. and Nielsen, R. (2016) SweepFinder2: increased robustness and flexibility. *Bioinformatics* **32**:1895-1897

Pavlidis, P., D. Zivkovic, A. Stamatakis, and N. Alachiotis, (2013) SweeD: likelihood-based detection of selective sweeps in thousands of genomes. *Molecular Biology and Evolution* **30**: 2224-34.

See Also

haplohh-class, data2haplohh

Examples

```
#example
# sweepfinder example from vignette
make.example.files()
hh <- data2haplohh("example_sweep_with_recombination.vcf")
haplohh2sweepfinder(hh)
remove.example.files()
```

---

haplohh_cgu_bta12 *Example of an* haplohh *object*

Description

The object contains haplotype data for 140 cattle individuals (280 haplotypes) belonging to the Creole breed from Guadeloupe (CGU) and 1424 markers (mapping to chromosome BTA12).

Usage

```
data(haplohh_cgu_bta12)
```

Format

An object of haplohh-class.

References

Gautier, M. and Naves, M. (2011). Footprints of selection in the ancestral admixture of a New World Creole cattle breed. *Molecular Ecology*, **20**, 3128-3143.

See Also

data2haplohh

ies2xpehh *Compute XP-EHH*

Description

Compute XP-EHH (standardized ratio of iES of two populations).

Usage

```
ies2xpehh(
  scan_pop1,
  scan_pop2,
  popname1 = NA,
  popname2 = NA,
  min_nhaplo = NA,
  standardize = TRUE,
  include_freq = FALSE,
  p.side = NA,
  p.adjust.method = "none",
  verbose = TRUE
)
```

Arguments

scan_pop1	a data frame with markers in rows and columns with chromosome name, position of the marker, frequency of the ancestral allele and iES as obtained by scan_hh on the first population.
scan_pop2	a data frame with markers in rows and columns with chromosome name, position of the marker, frequency of the ancestral allele and iES as obtained by scan_hh on the second population.
popname1	short ID/name of the first population; to be added to an output column name.
popname2	short ID/name of the second population; to be added to an output column name.
min_nhaplo	discard positions where in at least one of the populations fewer than min_nhaplo haplotypes have been evaluated (default NA).
standardize	logical. If TRUE (default), then standardize XP-EHH, else report unstandardized XP-EHH.
include_freq	logical. If TRUE include columns with allele frequencies into result.
p.side	side to which refers the p-value. Default NA, meaning two-sided. Can be set to "left" or "right".
p.adjust.method	method passed to function p.adjust to correct the p-value for multiple testing. Default "none".
verbose	logical. If TRUE (default), report number of markers of the two source data frames and result data frame.

Details

Log ratio of iES (population 1 over population 2) computed as described in Sabeti et al. (2007). Note that the two data frames are merged on the basis of chromosome and position. Marker names are kept, if they are identical and unique in both data frames.

Since the standardized XP-EHH values follow, if markers evolve predominantly neutrally, approximately a standard Gaussian distribution, it is practical to assign to the values a p-value relative to the null-hypothesis of neutral evolution. The parameter `p.side` determines if the p-value is assigned to both sides of the distribution or to one side of interest.

Value

The returned value is a data frame with markers in rows and columns for chromosome name, marker position, XP-EHH and, if standardized, p-value in a negative log10 scale. Optionally, allele frequencies are included.

References

Gautier, M. and Naves, M. (2011). Footprints of selection in the ancestral admixture of a New World Creole cattle breed. *Molecular Ecology*, **20**, 3128-3143.

Sabeti, P.C. et al. (2007). Genome-wide detection and characterization of positive selection in human populations. *Nature*, **449**, 913-918.

See Also

`scan_hh`, `distribplot`, `manhattanplot`

Examples

```
library(rehh.data)
data(wgscan.cgu) ; data(wgscan.eut)
## results from a genome scan (44,057 SNPs)
##see ?wgscan.eut and ?wgscan.cgu for details
wgscan.xpehh <- ies2xpehh(wgscan.cgu, wgscan.eut, "CGU", "EUT")
```

`ihh2ihs` *Compute iHS*

Description

Compute iHS (standardized ratio of iHH values of two alleles).

Usage

```
ihh2ihs(
  scan,
  freqbin = 0.025,
  min_maf = 0.05,
  min_nhaplo = NA,
  standardize = TRUE,
  include_freq = FALSE,
  right = FALSE,
  alpha = 0.05,
  p.side = NA,
  p.adjust.method = "none",
  verbose = TRUE
)
```

Arguments

scan	a data frame with chromosome name, marker position, frequency of ancestral (resp. major) allele, frequency of derived (resp. minor) allele, and iHH for both alleles, as obtained from function scan_hh.
freqbin	size of the bins to standardize log(iHH_A/iHH_D). Markers are binned with respect to the derived allele frequency at the focal marker. The bins are built from min_maf to 1-min_maf in steps of size freqbin. If set to 0, standardization is performed considering each observed frequency as a discrete frequency class (useful in case of a large number of markers and few different haplotypes). If set to an integer of 1 or greater, a corresponding number of equally sized bins are created.
min_maf	focal markers with a MAF (Minor Allele Frequency) lower than or equal to min_maf are discarded from the analysis (default 0.05).
min_nhaplo	focal markers with least one of the two compared alleles carried by fewer than min_nhaplo haplotypes, are discarded (default NA).
standardize	logical. If TRUE (default), then standardize iHS, else report unstandardized iHS.
include_freq	logical. If TRUE include columns with allele frequencies into result.
right	logical. If TRUE the bin intervals are closed on the right (and open on the left).
alpha	calculate quantiles alpha/2 and (1-alpha/2) for unstandardized binned iHS.
p.side	side to which refers the p-value. Default NA, meaning two-sided. Can be set to "left" or "right".
p.adjust.method	method passed to function p.adjust to correct the p-value for multiple testing. Default "none".
verbose	logical. If TRUE (default), report number of markers of the source data frame and result data frame.

Details

Computes log ratio of iHH of two focal alleles as described in Voight et al. (2006). The standardization is performed within each bins separately because of the frequency-dependence of expected iHS values under neutrality. An implicit assumption of this approach is that each bin is dominated by neutral markers.

Since the standardized iHS values follow, if markers evolve predominantly neutrally, approximately a standard Gaussian distribution, it is practical to assign to the values a p-value relative to the null-hypothesis of neutral evolution. The parameter `p.side` determines if the p-value is assigned to both sides of the distribution or to one side of interest.

Value

The returned value is a list containing two elements

ihs a data frame with markers in rows and the columns for chromosome name, marker position, iHS and, if standardized, p-value in a negative log10 scale. Optionally, allele frequencies are included.

frequency.class a data frame with bins in rows and columns for the number of markers, mean uniHS, standard deviation uniHS, lower quantile uniHS, upper quantile uniHS.

References

Gautier, M. and Naves, M. (2011). Footprints of selection in the ancestral admixture of a New World Creole cattle breed. *Molecular Ecology*, **20**, 3128-3143.

Voight, B.F. and Kudaravalli, S. and Wen, X. and Pritchard, J.K. (2006). A map of recent positive selection in the human genome. *Plos Biology*, **4**, e72.

See Also

`scan_hh`, `distribplot`, `freqbinplot`, `manhattanplot`

Examples

```
library(rehh.data)
data(wgscan.cgu)
#results from a genome scan (44,057 SNPs)
#see ?wgscan.eut and ?wgscan.cgu for details
wgscan.cgu.ihs <- ihh2ihs(wgscan.cgu)
```

`ines2rsb` *Compute Rsb*

Description

Compute Rsb (standardized ratio of inES of two populations).

Usage

```
ines2rsb(
  scan_pop1,
  scan_pop2,
  popname1 = NA,
  popname2 = NA,
  min_nhaplo = NA,
  standardize = TRUE,
  include_freq = FALSE,
  p.side = NA,
  p.adjust.method = "none",
  verbose = TRUE
)
```

Arguments

scan_pop1	a data frame with markers in rows and columns with chromosome name, position of the marker, frequency of the ancestral allele and inES as obtained by scan_hh on the first population.
scan_pop2	a data frame with markers in rows and columns with chromosome name, position of the marker, frequency of the ancestral allele and inES as obtained by scan_hh on the second population.
popname1	short ID/name of the first population; to be added to an output column name.
popname2	short ID/name of the second population; to be added to an output column name.
min_nhaplo	discard positions where in at least one of the populations fewer than min_nhaplo haplotypes have been evaluated (default NA).
standardize	logical. If TRUE (default), then standardize Rsb, else report unstandardized Rsb.
include_freq	logical. If TRUE include columns with allele frequencies into result.
p.side	side to which refers the p-value. Default NA, meaning two-sided. Can be set to "left" or "right".
p.adjust.method	method passed to function p.adjust to correct the p-value for multiple testing. Default "none".
verbose	logical. If TRUE (default), report number of markers of the two source data frames and result data frame.

Details

Log ratio of inES (population 1 over population 2) computed as described in Tang et al. (2007). Note that the two data frames are merged on the basis of chromosome and position. Marker names are kept, if they are identical and unique in both data frames.

Since the standardized Rsb values follow, if markers evolve predominantly neutrally, approximately a standard Gaussian distribution, it is practical to assign to the values a p-value relative to the null-hypothesis of neutral evolution. The parameter p.side determines if the p-value is assigned to both sides of the distribution or to one side of interest.

Value

The returned value is a data frame with markers in rows and columns for chromosome name, marker position, Rsb and, if standardized, p-value in a negative log10 scale. Optionally, allele frequencies are included.

References

Gautier, M. and Naves, M. (2011). Footprints of selection in the ancestral admixture of a New World Creole cattle breed. *Molecular Ecology*, **20**, 3128-3143.

Tang, K. and Thornton, K.R. and Stoneking, M. (2007). A New Approach for Using Genome Scans to Detect Recent Positive Selection in the Human Genome. *Plos Biology*, **7**, e171.

See Also

`scan_hh`, `distribplot`, `manhattanplot`

Examples

```
library(rehh.data)
data(wgscan.cgu) ; data(wgscan.eut)
## results from a genome scan (44,057 SNPs)
##see ?wgscan.eut and ?wgscan.cgu for details
wgscan.rsb <- ines2rsb(wgscan.cgu, wgscan.eut, "CGU", "EUT")
```

`make.example.files` *Copy example input files into current working directory*

Description

This function copies the following example files to the current working directory:

- `example1.hap` "example 1" haplotype file in "standard format"
- `example1.map` "example 1" marker information file
- `example1.vcf` "example 1" as vcf file
- `example2.hap` "example 2" haplotype file in "standard format"
- `example2.map` "example 2" marker information file
- `example2.vcf` "example 2" as vcf file
- `example_neutral.vcf` "example neutral evolution" as vcf file
- `example_sweep.vcf` "example for a selective sweep (without recombination)"
- `example_sweep_with_recombination.vcf` "example for a selective sweep with recombination
- `ms.out` output from a small simulation by the program 'ms'
- `bta12_cgu.hap` an haplotype file in "standard format"
- `bta12_cgu.thap` an haplotype file in "transposed format"

- `bta12_hapguess_switch.out` an haplotype file in fastphase output format
- `map.inp` a marker information file for all bta_cgu markers

Example 1 was used in (Gautier 2017) to explain the various EHH derived statistics calculated by this package. Example 2 is an extension containing multi-allelic markers and missing values.

Examples for neutral data and sweeps are discussed in a supplement of Klassmann (2020).

The bta12 files contain data for 280 haplotypes, originating from 140 individuals belonging to the Creole cattle breed from Guadeloupe, at 1.424 markers mapping to bovine chromosome 12 (BTA12) (Gautier 2011).

Usage

```
make.example.files()
```

References

Gautier, M. and Naves, M. (2011). Footprints of selection in the ancestral admixture of a New World Creole cattle breed. *Molecular Ecology*, **20**, 3128-3143.

Gautier, M., Klassmann, A. and Vitalis, R. (2017). rehh 2.0: a reimplementation of the R package rehh to detect positive selection from haplotype structure. *Molecular Ecology Resources*, **17**, 78-90.

Klassmann, A. and Gautier, M. (2022). Detecting selection using extended haplotype homozygosity (EHH)-based statistics in unphased or unpolarized data. *PLoS One*. **17**(1):e0262024 doi:10.1371/journal.pone.0262024

See Also

`data2haplohh`, `remove.example.files`

`manhattanplot` *Manhattan plot of iHS, XP-EHH or Rsb over a genome.*

Description

Manhattanplot of iHS, XP-EHH or Rsb over a genome.

Usage

```
manhattanplot(
  data,
  pval = FALSE,
  threshold = c(-2, 2),
  chr.name = NA,
  cr = NULL,
  cr.col = "gray",
  cr.opacity = 0.5,
  cr.lab.cex = 0.6,
```

```
  cr.lab.offset = 0,
  cr.lab.pos = "top",
  mrk = NULL,
  mrk.cex = 1,
  mrk.col = "gray",
  mrk.pch = 1,
  mrk.lab.cex = 0.4,
  mrk.lab.pos = 4,
  ignore_sign = FALSE,
  cex = 0.5,
  las = 1,
  pch = 20,
  inset = 5e+06,
  resolution = NULL,
  ...
)
```

Arguments

data	output of either ihh2ihs, ies2xpehh or ines2rsb.
pval	logical. If TRUE, the p-value is plotted, otherwise the score itself.
threshold	a horizontal line is added at the corresponding value(s), for instance to represent a significance threshold. A single value (upper or lower threshold) or two values (upper and lower) can be specified.
chr.name	if NA (default), all chromosomes are plotted, otherwise only those specified.
cr	highlight "candidate regions" specified by a data.frame with columns CHR, START and END as obtained by the function calc_candidate_regions.
cr.col	the color for highlighting
cr.opacity	a value between 0 (invisible) and 1 (opaque).
cr.lab.cex	text size of candidate region labels.
cr.lab.offset	offset of candidate region labels.
cr.lab.pos	if "top" (default) or "bottom", candidate regions are labeled by numbers; to turn off, use "none"
mrk	highlight marker specified by a data.frame containing the colums CHR and POSITION. The row names of that data frame are taken as labels. Alternatively a vector with marker IDs can be specified. In the latter case the ID is used as label.
mrk.cex	size of marker label.
mrk.col	color of the highlighted points.
mrk.pch	type of the highlighted points.
mrk.lab.cex	text size of marker label. If zero, no labels are printed.
mrk.lab.pos	a position specifier for the text. Values of 1, 2, 3 and 4, respectively indicate positions below, to the left of, above and to the right of the highlighted marker.
ignore_sign	logical. If TRUE, absolute values are plotted.
cex	size of the points representing markers in the plot(s) (see par).

las	orientation of axis labels (see par).
pch	type of the points representing markers in the plot(s) (see points).
inset	inset (in bases) between chromosomes to avoid overlap of data points. Default: 5,000,000 bases.
resolution	Rasterize data points to the specified resolution and remove duplicate points. Defaults to NULL, i.e. no rasterization. A typical value might be c(1E5, 0.01), meaning that resolution on the x-axis (chromosomal position) is 100000 and on the y-axis (score or p-value) is 0.01.
...	further arguments to be passed to plot.default.

Details

The color of chromosomes is taken from the "Graphics Palette", see palette.

If a single chromosome is plotted, a genomic region can be specified by argument xlim.

Other statistics can be plotted as well, although a warning is issued. They must be given by a data.frame with columns CHR and POSITION and the statistic in the third column.

Value

The function returns a plot.

See Also

ihh2ihs, ies2xpehh, ines2rsb, calc_candidate_regions.

Examples

```
library(rehh.data)
data(wgscan.cgu)
## results from a genome scan (44,057 SNPs)
## see ?wgscan.eut and ?wgscan.cgu for details
wgscan.ihs <- ihh2ihs(wgscan.cgu)
manhattanplot(wgscan.ihs)
```

---

plot.ehh *Plot EHH around a focal marker*

Description

Plot curve of EHH values around a focal marker.

Usage

```
## S3 method for class 'ehh'
plot(
  x,
  ylim = c(0, 1),
  type = "l",
  main = paste0("EHH around '", x$mrk.name, "'"),
  xlab = "Position",
  ylab = "Extended Haplotype Homozygosity",
  col = c("blue", "red", "violet", "orange"),
  mrk.col = "gray",
  bty = "n",
  legend = NA,
  legend.xy.coords = "automatic",
  ...
)
```

Arguments

x	data (output of calc_ehh).
ylim	the y limits of the plot
type	plot type (see plot.default and matplot). Type "s" or "S" yield both (the same) piecewise constant curve.
main	title for the plot (default NA, i.e. none).
xlab	title for the x-axis.
ylab	title for the y-axis.
col	color for the ancestral and derived alleles (respectively) curves.
mrk.col	color of the vertical line at the focal marker position.
bty	box type around plot (see par).
legend	legend text.
legend.xy.coords	if "automatic" (default) places legend either top left or top right; if "none", no legend is drawn; otherwise the argument is passed to legend.
...	further arguments to be passed to function matplot.

See Also

data2haplohh, calc_ehh, plot.ehhs, scan_hh.

Examples

```
#example haplohh object (280 haplotypes, 1424 SNPs)
#see ?haplohh_cgu_bta12 for details
data(haplohh_cgu_bta12)
#computing EHH statistics for the marker "F1205400"
#which displays a strong signal of selection
```

```
ehh <- calc_ehh(haplohh_cgu_bta12, mrk = "F1205400")
plot(ehh)
```

`plot.ehhs` *Plot EHHS around a focal marker*

Description

Plot curve of EHHS values around a focal marker.

Usage

```
## S3 method for class 'ehhs'
plot(
  x,
  nehhs = FALSE,
  ylim = c(0, 1),
  type = "l",
  main = paste0("EHHS around '", x$mrk.name, "'"),
  xlab = "Position",
  ylab = "Extended Haplotype Homozygosity per Site",
  bty = "n",
  mrk.col = "gray",
  ...
)
```

Arguments

`x`	data (output of `calc_ehhs`).
`nehhs`	logical. If `TRUE`, plot normalized EHHS.
`ylim`	the y limits of the plot
`type`	plot type (see `plot.default`. Type `"s"` or `"S"` yield both (the same) piecewise constant curve.
`main`	title for the plot (default `NA`, i.e. none).
`xlab`	title for the x-axis.
`ylab`	title for the y-axis.
`bty`	box type around plot (see `par`).
`mrk.col`	color of the vertical line at the focal marker position.
`...`	further arguments to be passed to function `plot.default`.

See Also

`data2haplohh`, `plot.ehh`, `calc_ehhs`, `scan_hh`.

Examples

```
#example haplohh object (280 haplotypes, 1424 SNPs)
#see ?haplohh_cgu_bta12 for details
data(haplohh_cgu_bta12)
#computing EHHS statisitics for the marker "F1205400"
#which displays a strong signal of selection
ehhs <- calc_ehhs(haplohh_cgu_bta12, mrk = "F1205400")
plot(ehhs)
```

`plot.furcation` *Plots furcation trees around a focal marker*

Description

Plots furcation trees around a focal marker

Usage

```
## S3 method for class 'furcation'
plot(
  x,
  allele = NA,
  col = c("blue", "red", "violet", "orange"),
  mrk.col = "gray",
  lwd = 0.1,
  hap.names = NULL,
  cex.lab = 1,
  family.lab = "",
  offset.lab = 0.5,
  legend = NA,
  legend.xy.coords = "automatic",
  ...
)
```

Arguments

`x`	an object of class furcation (see `calc_furcation`).
`allele`	If `NA` (default), furcation trees for all alleles of the focal marker are plotted, otherwise for the specified alleles. Alleles must be specified by their internal coding, i.e. '0' for ancestral resp. major allele, etc.
`col`	color for each allele (as coded internally).
`mrk.col`	color of the vertical line at the focal marker position.
`lwd`	controls the relative width of the diagram lines on the plot (default 0.1).
`hap.names`	a vector containing names of chromosomes.
`cex.lab`	relative size of labels. See `par`.

family.lab	font family for labels. See par.
offset.lab	offset of labels. See par.
legend	legend text.
legend.xy.coords	if "automatic" (default) places legend either top left or top right; if "none", no legend is drawn; otherwise argument is passed to legend.
...	other arguments to be passed to plot.default.

See Also

plot.haplen.

Examples

```
#example haplohh object (280 haplotypes, 1424 SNPs)
#see ?haplohh_cgu_bta12 for details
data(haplohh_cgu_bta12)
#plotting furcation diagram for both ancestral and derived allele
#from the marker "F1205400"
#which display a strong signal of selection
f <- calc_furcation(haplohh_cgu_bta12, mrk = "F1205400")
plot(f)
plot(f, xlim = c(2e+07,3.5e+07))
plot(f, xlim = c(2.7e+07,3.1e+07))
plot(f, xlim = c(2.7e+07,3.1e+07), hap.names = hap.names(haplohh_cgu_bta12), cex.lab=0.3)
```

---

plot.haplen *Plot the length of extended haplotypes around a focal marker*

Description

Plot the length of extended haplotype around a focal marker.

Usage

```
## S3 method for class 'haplen'
plot(
  x,
  allele = NA,
  group_by_allele = TRUE,
  order_by_length = FALSE,
  col = c("blue", "red", "violet", "orange"),
  mrk.col = "gray",
  lwd = 1,
  hap.names = NULL,
  cex.lab = 1,
  family.lab = "",
```

```
  offset.lab = 0.5,
  pos.lab = "left",
  legend = NA,
  legend.xy.coords = "automatic",
  ...
)
```

Arguments

x	an object of class haplen generated by calc_haplen.
allele	if NA (default), haplotypes of all alleles are plotted, otherwise for the specified alleles. Alleles must be specified by their internal coding, i.e. '0' for ancestral resp. major allele, etc.
group_by_allele	logical. If TRUE (default), group chromosomes by their allele at the focal marker; alleles are ordered by their internal coding unless parameter alleles is specified. If FALSE, haplotypes are drawn by their order in the input file.
order_by_length	if TRUE, chromosomes are ordered by their shared haplotype length.
col	color for each allele (as coded internally).
mrk.col	color of the vertical line at the focal marker position.
lwd	line width.
hap.names	a vector containing the names of chromosomes.
cex.lab	relative letter size of labels. See par.
family.lab	font family for labels. See par.
offset.lab	offset of labels. See par.
pos.lab	position of haplotype labels. Either "left" , "right" or "both".
legend	legend text.
legend.xy.coords	if "automatic" (default) places legend either top left or top right; if "none", no legend is drawn; otherwise argument is passed to legend.
...	other parameters to be passed to plot.default.

See Also

calc_haplen, plot.furcation.

Examples

```
#example haplohh object (280 haplotypes, 1424 SNPs)
#see ?haplohh_cgu_bta12 for details
data(haplohh_cgu_bta12)
#plotting length of extended haplotypes for both ancestral and derived allele
#of the marker "F1205400"
#which displays a strong signal of selection
f <- calc_furcation(haplohh_cgu_bta12, mrk = "F1205400")
```

```
h <- calc_haplen(f)
plot(h)
plot(h, hap.names = hap.names(haplohh_cgu_bta12), cex.lab = 0.3)
```

`plot.haplohh` *Plot the variants of a haplohh object*

Description

Plot the variants of a haplohh object. This method is intended for visualization of very small data sets such as the examples provided by the package.

Usage

```
## S3 method for class 'haplohh'
plot(
  x,
  mrk = NA,
  allele = NA,
  group_by_allele = FALSE,
  ignore.distances = FALSE,
  col = c("blue", "red", "violet", "orange"),
  linecol = "gray",
  mrk.col = "gray",
  pch = 19,
  cex = 1,
  lwd = 1,
  hap.names = NULL,
  mrk.names = NULL,
  cex.lab.hap = 0.8,
  cex.lab.mrk = 0.8,
  family.lab = "",
  offset.lab.hap = 0.5,
  offset.lab.mrk = 0.25,
  pos.lab.hap = "left",
  pos.lab.mrk = "top",
  srt.hap = 0,
  srt.mrk = 0,
  highlight.mrk = NULL,
  highlight.mrk.col = c("lightgray", "black", "darkgray"),
  ...
)
```

Arguments

`x`	an object of class `haplo-hh` generated by `data2haplohh`.
`mrk`	the focal marker. Used only, if alleles are grouped or (de-)selected.

allele	if NA (default), haplotypes of all alleles are plotted, otherwise for the specified alleles. Alleles must be specified by their internal coding, i.e. '0' for ancestral resp. major allele, etc. Haplotypes with missing values at the focal marker can only be selected in combination with genotyped alleles, e.g. as c(1,NA).
group_by_allele	logical. If TRUE, group chromosomes by their allele at the focal marker; alleles are ordered by their internal coding unless parameter alleles is specified. If FALSE (default), haplotypes are drawn by their order in the input file.
ignore.distances	logical. If TRUE, markers are drawn equally-spaced.
col	color for each allele (as coded internally).
linecol	the color of the background lines. If more than one color is specified and sequences sorted by the marker allele, the specified colors are used to distinguish the alleles; otherwise consecutive sequences are set into the specified colors.
mrk.col	color of the vertical line at the focal marker position.
pch	symbol used for markers. See points.
cex	relative size of marker symbol. See points.
lwd	line width.
hap.names	a vector containing the names of chromosomes.
mrk.names	a vector containing the names of markers.
cex.lab.hap	relative letter size of haplotype labels. See par.
cex.lab.mrk	relative letter size of marker labels. See par.
family.lab	font family for labels. See par.
offset.lab.hap	offset of haplotype labels. See par.
offset.lab.mrk	offset of marker labels. See par.
pos.lab.hap	position of haplotype labels. Either "left" (default), "right", "none" or "both".
pos.lab.mrk	position of marker labels. Either "top" (default) or "none".
srt.hap	rotation of haplotype labels (see par).
srt.mrk	rotation of marker labels (see par).
highlight.mrk	vector of markers to be highlighted
highlight.mrk.col	color for each allele (as coded internally) at highlighted markers.
...	other parameters to be passed to plot.default.

Details

Specifying a haplohh-object with more than 4096 haplotypes or markers produces an error.

See Also

calc_haplen, plot.furcation.

Examples

```
#example haplohh object
make.example.files()
hh <- data2haplohh(hap_file = "example1.hap",
                   map_file = "example1.map",
                   allele_coding = "01")
plot(hh)
hh <- data2haplohh(hap_file = "example2.hap",
                   map_file = "example2.map",
                   allele_coding = "01",
                   min_perc_geno.mrk = 50)
plot(hh)
remove.example.files()
```

---

remove.example.files *Remove example files from current working directory.*

Description

Remove example files from current working directory.

Usage

```
remove.example.files()
```

Details

Removes the files created by `make.example.files()`. No error is thrown, if files do not exist.

See Also

`make.example.files`

---

scan_hh *Compute iHH, iES and inES over a whole chromosome*

Description

Compute integrated EHH (iHH), integrated EHHS (iES) and integrated normalized EHHS (inES) for all markers of a chromosome (or linkage group).

Usage

```
scan_hh(
  haplohh,
  limhaplo = 2,
  limhomohaplo = 2,
  limehh = 0.05,
  limehhs = 0.05,
  phased = TRUE,
  polarized = TRUE,
  scalegap = NA,
  maxgap = NA,
  discard_integration_at_border = TRUE,
  lower_ehh_y_bound = limehh,
  lower_ehhs_y_bound = limehhs,
  interpolate = TRUE,
  threads = 1
)
```

Arguments

haplohh	an object of class haplohh (see data2haplohh)
limhaplo	if there are less than limhaplo chromosomes that can be used for the calculation of EHH(S), the calculation is stopped. The option is intended for the case of missing data, which leads to the successive exclusion of haplotypes: the further away from the focal marker the less haplotypes contribute to EHH(S).
limhomohaplo	if there are less than limhomohaplo homozygous chromosomes, the calculation is stopped. This option is intended for unphased data and should be invoked only if relatively low frequency variants are not filtered subsequently (see main vignette and Klassmann et al. 2020).
limehh	limit at which EHH stops to be evaluated.
limehhs	limit at which EHHS stops to be evaluated.
phased	logical. If TRUE (default) chromosomes are expected to be phased. If FALSE, the haplotype data is assumed to consist of pairwise ordered chromosomes belonging to diploid individuals. EHH(S) is then estimated over individuals which are homozygous at the focal marker.
polarized	logical. TRUE by default. If FALSE, use major and minor allele instead of ancestral and derived. If there are more than two alleles then the minor allele refers to the second-most frequent allele.
scalegap	scale or cap gaps larger than the specified size to the specified size (default=NA, i.e. no scaling).
maxgap	maximum allowed gap in bp between two markers. If exceeded, further calculation of EHH(S) is stopped at the gap (default=NA, i.e no limitation).
discard_integration_at_border	logical. If TRUE (default) and computation reaches first or last marker or a gap larger than maxgap, iHH, iES and inES are set to NA.

`lower_ehh_y_bound`
: lower y boundary of the area to be integrated over (default: `limehh`). Can be set to zero for compatibility with the program hapbin.

`lower_ehhs_y_bound`
: lower y boundary of the area to be integrated (default: `limehhs`). Can be set to zero for compatibility with the program hapbin.

`interpolate`
: logical. If `TRUE` (default), integration is performed over a continuous EHH(S) curve (values are interpolated linearly between consecutive markers), otherwise the EHH(S) curve decreases stepwise at markers.

`threads`
: number of threads to parallelize computation

Details

Integrated EHH (iHH), integrated EHHS (iES) and integrated normalized EHHS (inES) are computed for all markers of the chromosome (or linkage group). This function is several times faster as a procedure calling in turn `calc_ehh` and `calc_ehhs` for all markers. To perform a whole genome-scan this function needs to be called for each chromosome and results concatenated.

Note that setting `limehh` or `limehhs` to zero is likely to reduce power, since even under neutrality a tiny fraction («0.05) of extremely long shared haplotypes is expected which, if fully accounted for, would obfuscate the signal at selected sites.

Value

The returned value is a dataframe with markers in rows and the following columns

1. chromosome name
2. position in the chromosome
3. sample frequency of the ancestral / major allele
4. sample frequency of the second-most frequent remaining allele
5. number of evaluated haplotypes at the focal marker for the ancestral / major allele
6. number of evaluated haplotypes at the focal marker for the second-most frequent remaining allele
7. iHH of the ancestral / major allele
8. iHH of the second-most frequent remaining allele
9. iES (used by Sabeti et al 2007)
10. inES (used by Tang et al 2007)

Note that in case of unphased data the evaluation is restricted to haplotypes of homozygous individuals which reduces the power to detect selection, particularly for iHS (for appropriate parameter setting see the main vignette and Klassmann et al (2020)).

References

Gautier, M. and Naves, M. (2011). Footprints of selection in the ancestral admixture of a New World Creole cattle breed. *Molecular Ecology*, **20**, 3128-3143.

Klassmann, A. and Gautier, M. (2022). Detecting selection using extended haplotype homozygosity (EHH)-based statistics in unphased or unpolarized data. *PLoS One*. **17**(1):e0262024 doi:10.1371/journal.pone.0262024

Sabeti, P.C. et al. (2002). Detecting recent positive selection in the human genome from haplotype structure. *Nature*, **419**, 832-837.

Sabeti, P.C. et al. (2007). Genome-wide detection and characterization of positive selection in human populations. *Nature*, **449**, 913-918.

Tang, K. and Thornton, K.R. and Stoneking, M. (2007). A New Approach for Using Genome Scans to Detect Recent Positive Selection in the Human Genome. *Plos Biology*, **7**, e171.

Voight, B.F. and Kudaravalli, S. and Wen, X. and Pritchard, J.K. (2006). A map of recent positive selection in the human genome. *Plos Biology*, **4**, e72.

See Also

data2haplohh, calc_ehh, calc_ehhs ihh2ihs,ines2rsb, ies2xpehh

Examples

```
#example haplohh object (280 haplotypes, 1424 SNPs)
#see ?haplohh_cgu_bta12 for details
data(haplohh_cgu_bta12)
scan <- scan_hh(haplohh_cgu_bta12)
```

---

scan_hh_full *Compute iHH, iES and inES over a whole chromosome without cut-offs*

Description

Compute integrated EHH (iHH), integrated EHHS (iES) and integrated normalized EHHS (inES) for all markers of a chromosome (or linkage group). This function computes the statistics by a slightly different algorithm than scan_hh: it sidesteps the calculation of EHH and EHHS values and their subsequent integration and consequently no cut-offs relying on these values can be specified. Instead, it computes the (full) lengths of pairwise shared haplotypes and averages them afterwords.

This function is primarily intended for the study of general properties of these statistics using simulated data.

Usage

```
scan_hh_full(
  haplohh,
  phased = TRUE,
  polarized = TRUE,
  maxgap = NA,
  max_extend = NA,
  discard_integration_at_border = TRUE,
```

```
  geometric.mean = FALSE,
  threads = 1
)
```

Arguments

haplohh	an object of class haplohh (see data2haplohh)
phased	logical. If TRUE (default) chromosomes are expected to be phased. If FALSE, the haplotype data is assumed to consist of pairwise ordered chromosomes belonging to diploid individuals. EHH(S) is then estimated over individuals which are homozygous at the focal marker.
polarized	logical. TRUE by default. If FALSE, use major and minor allele instead of ancestral and derived. If there are more than two alleles then the minor allele refers to the second-most frequent allele.
maxgap	maximum allowed gap in bp between two markers. If exceeded, further calculation of EHH(S) is stopped at the gap (default=NA, i.e. no limitation).
max_extend	maximum distance in bp to extend shared haplotypes away from the focal marker. (default NA, i.e. no limitation).
discard_integration_at_border	logical. If TRUE (default) and computation of any of the statistics reaches first or last marker or a gap larger than maxgap, iHH, iES and inES are set to NA.
geometric.mean	logical. If FALSE (default), the standard arithmetic mean is used to average shared haplotype lengths. If TRUE the geometric mean is used instead (IES values are undefined in this case). Note that usage of the geometric mean has not yet been studied formally and should be considered experimental!
threads	number of threads to parallelize computation

Details

Integrated EHH (iHH), integrated EHHS (iES) and integrated normalized EHHS (inES) are computed for all markers of the chromosome (or linkage group). This function sidesteps the computation of EHH and EHHS values and their stepwise integration. Instead, the length of all shared haplotypes is computed and afterwords averaged. In the absence of missing values the statistics are identical to those calculated by scan_hh with settings limehh = 0, limehhs = 0, lower_ehh_y_bound = 0 and interpolate = FALSE, yet this function is faster.

Application of a cut-off is necessary for reducing the spurious signals of selection caused by single shared haplotypes of extreme length. Hence, e.g. for human experimental data it might be reasonable to set max_extend to 1 or 2 Mb.

scan_hh computes the statistics iHH_A, ihh_D and iES/inES separately, while this function calculates them simultaneously. Hence, if discard_integration_at_border is set to TRUE and the extension of shared haplotypes reaches a border (i.e. chromosomal boundaries or a gap larger than maxgap), this function discards all statistics.

The handling of missing values is different, too: scan_hh "removes" chromosomes with missing values from further calculations. EHH and EHHS are then calculated for the remaining chromosomes which can accidentally yield an increase in EHH or EHHS. This can not happen with

`scan_hh_full()` which treats each missing value of a marker as if it were a new allele - terminating any shared haplotype, but does changing the set of considered chromosomes. Thus, missing values cause a faster decay of EHH(S) with function `scan_hh_full()`.

Value

The returned value is a dataframe with markers in rows and the following columns

1. chromosome name
2. position in the chromosome
3. sample frequency of the ancestral / major allele
4. sample frequency of the second-most frequent remaining allele
5. number of evaluated haplotypes at the focal marker for the ancestral / major allele
6. number of evaluated haplotypes at the focal marker for the second-most frequent remaining allele
7. iHH of the ancestral / major allele
8. iHH of the second-most frequent remaining allele
9. iES (used by Sabeti et al 2007)
10. inES (used by Tang et al 2007)

Note that in case of unphased data the evaluation is restricted to haplotypes of homozygous individuals which reduces the power to detect selection, particularly for iHS (for appropriate parameter setting see the main vignette and Klassmann et al (2020)).

References

Gautier, M. and Naves, M. (2011). Footprints of selection in the ancestral admixture of a New World Creole cattle breed. *Molecular Ecology*, **20**, 3128-3143.

Klassmann, A. and Gautier, M. (2022). Detecting selection using extended haplotype homozygosity (EHH)-based statistics in unphased or unpolarized data. *PLoS One*. **17**(1):e0262024 doi:10.1371/journal.pone.0262024

Sabeti, P.C. et al. (2002). Detecting recent positive selection in the human genome from haplotype structure. *Nature*, **419**, 832-837.

Sabeti, P.C. et al. (2007). Genome-wide detection and characterization of positive selection in human populations. *Nature*, **449**, 913-918.

Tang, K. and Thornton, K.R. and Stoneking, M. (2007). A New Approach for Using Genome Scans to Detect Recent Positive Selection in the Human Genome. *Plos Biology*, **7**, e171.

Voight, B.F. and Kudaravalli, S. and Wen, X. and Pritchard, J.K. (2006). A map of recent positive selection in the human genome. *Plos Biology*, **4**, e72.

See Also

`data2haplohh`, `scan_hh`, `ihh2ihs`, `ines2rsb`, `ies2xpehh`

Examples

```
#example haplohh object (280 haplotypes, 1424 SNPs)
#see ?haplohh_cgu_bta12 for details
data(haplohh_cgu_bta12)
#using function scan_hh() with no cut-offs
scan <- scan_hh(haplohh_cgu_bta12, discard_integration_at_border = FALSE,
limehh = 0, limehhs = 0, lower_ehh_y_bound = 0, interpolate = FALSE)
#using function scan_hh_full()
scan_full <- scan_hh_full(haplohh_cgu_bta12, discard_integration_at_border = FALSE)
#both yield identical results within numerical precision
all.equal(scan, scan_full)
#calculate iHH "by hand"
m <- calc_pairwise_haplen(haplohh_cgu_bta12, mrk = "F1205400")
#extract alleles of focal marker
foc_alleles <- haplo(haplohh_cgu_bta12)[, "F1205400"]
#subset matrix to contain only ancestral or derived pairwise haplotype lengths
m_A <- m[foc_alleles == 0, foc_alleles == 0]
m_D <- m[foc_alleles == 1, foc_alleles == 1]
#calculate mean values (the matrices are symmetric and their diagonal is zero)
mean(m_A[upper.tri(m_A)])
mean(m_D[upper.tri(m_D)])
# compare with IHH_A and IHH_D from full scan
scan_full["F1205400", ]
```

---

subset.haplohh *Subsets object of* haplohh-class

Description

Subsets the data of an object of class haplohh-class, meeting certain conditions.

Usage

```
## S3 method for class 'haplohh'
subset(
  x,
  select.hap = NULL,
  select.mrk = NULL,
  min_perc_geno.hap = NA,
  min_perc_geno.mrk = 100,
  min_maf = NA,
  max_alleles = NA,
  verbose = TRUE,
  ...
)
```

Arguments

x	object of class haplohh-class to be subset.
select.hap	expression, indicating haplotypes to select.
select.mrk	expression, indicating markers to select.
min_perc_geno.hap	threshold on percentage of missing data for haplotypes (haplotypes with less than min_perc_geno.hap percent of markers genotyped are discarded). Default is NA, hence no constraint.
min_perc_geno.mrk	threshold on percentage of missing data for markers (markers genotyped on less than min_perc_geno.mrk percent of haplotypes are discarded). By default, min_perc_geno.mrk=100, hence only fully genotyped markers are retained. This value cannot be set to NA or zero.
min_maf	threshold on the Minor Allele Frequency. Markers having a MAF lower than or equal to minmaf are discarded. In case of multi-allelic markers the second-most frequent allele is referred to as minor allele. Setting this value to zero eliminates monomorphic sites. Default is NA, hence no constraint.
max_alleles	threshold for the maximum number of different alleles at a site. Default is NA, hence no restriction. In order to retain only bi-allelic markers, set this parameter to 2.
verbose	logical. If TRUE (default), report verbose progress.
...	further arguments are ignored.

See Also

haplohh-class, data2haplohh

Examples

```
#example haplohh object (280 haplotypes, 1424 SNPs)
#see ?haplohh_cgu_bta12 for details
data(haplohh_cgu_bta12)
#select subset of first 10 hyplotypes and first 5 markers
subset(haplohh_cgu_bta12, select.hap = 1:10, select.mrk = 1:5)
```

---

update_haplohh *Update object of class haplohh*

Description

Update object of class haplohh-class constructed by rehh versions up to version 2.0.4.

Usage

```
update_haplohh(haplohh)
```

Arguments

haplohh	an object of an old version of haplohh-class.

Details

This function is intended to update haplohh objects that have been built by rehh versions up to 2.0.4. These objects cannot be used in functions of the current version. The following changes have been made to the class definition: The internal representation of the haplotype matrix followed the encoding

- 0 missing value
- 1 ancestral allele
- 2 derived allele

and has been replaced by a vcf-like encoding:

- NA missing value
- 0 ancestral allele
- 1 derived allele.

Furthermore the slots nsnp, snp.name and nhap have been removed and slot position renamed to positions. An update of an old haplohh object is done as follows:

new_haplohh = update_haplohh(old_haplohh).

See Also

haplohh-class, data2haplohh.

Index