Package 'genogeographer'

October 13, 2022

Type Package

Title Methods for Analysing Forensic Ancestry Informative Markers

Version 0.1.19

Author Torben Tvedebrink

Maintainer Torben Tvedebrink <tvede@math.aau.dk>

Depends R (>= 3.1.0)

Imports leaflet, shiny, shinyjs, knitr, DT, shinycssloaders, purrr, dplyr, magrittr, tidyr, ggplot2, tibble, forcats, readr, rmarkdown, rio, maps, shinyWidgets, rlang

Suggests tidyverse

Description Evaluates likelihood ratio tests for alleged ancestry. Implements the methods of Tvedebrink et al (2018) <doi:10.1016/j.tpb.2017.12.004>.

License GPL (>= 2)

Encoding UTF-8

LazyData true

RoxygenNote 6.1.1

NeedsCompilation no

Repository CRAN

Date/Publication 2019-09-27 10:20:08 UTC

R topics documented:

`app_genogeo` *Shiny application for GenoGeoGrapher*

Description

Shiny application for GenoGeoGrapher

Usage

```
app_genogeo(db_list = NULL, reporting_panel = TRUE)
```

Arguments

`db_list`	A named list of databases of reference populations. Each component is expected to be returned from `pops_to_DB`.
`reporting_panel`	Logical. Should report generate and download be available after sample analysis.

`bar_colour` *bar_colour*

Description

Creates the colour scale for the accepted and rejected populations based on z-score and the log likelihood (log P).

Usage

```
bar_colour(df, alpha = 1)
```

Arguments

`df`	A data.frame with at least three coloums. The first column is the logP, the second logical (z_score accept/reject), the third a unique naming column.
`alpha`	Should the alpha opacity be applied? And what value, 1 = solid, 0 = transparent.

error_bar_plot *Plot log likelihoods of profiles with approximate confidence intervals*

Description

Plots the estimated profile probabilities in each population. The colour depends on the profiles likelihood and rejection/acceptance (blue/red) based on z-score

Usage

```
error_bar_plot(data)
```

Arguments

data	The output from the genogeo function

Value

A barplot of the log likelihoods for each population with confidence limits

Author(s)

Torben Tvedebrink, <tvede@math.aau.dk>

Examples

```
df_ <- simulate_pops(pop_n = 20, aims_n = 50)
df_db <- pops_to_DB(df_)
profile <- random_AIMs_profile(df_db, keep_pop = TRUE)
profile$pop[1] # The true population
result <- genogeo(profile[,c("locus","x0")], df = df_db)
error_bar_plot(result)
```

exponent_tilt *P-values from Importing Sampling using Exponential tilting*

Description

P-values from Importing Sampling using Exponential tilting

Usage

```
exponent_tilt(x0, x1, n, p_limit = 0.1, B = 500, return_all = FALSE)
```

Arguments

x0	Allele count of profile
x1	Population allele count
n	Sampled alleles in total in population
p_limit	Upper limit to which we use the normal approximation
B	An integer specifying the number of importance samples.
return_all	Default is FALSE. If TRUE: Returns p-value, standard deviation, and method (and diagnostics).

Details

The method of importance sampling described in Tvedebrink et al (2018), Section 2.3 is implemented. It relies on exponential tilting of the proposal distribution using in the importance sampling.

Value

If return_all= FALSE the p-value is returned. Otherwise list of elements (see return_all) is returned.

---

genogeo *Likelihood ratio tests for AIMs*

Description

Computes the likelihood ratio test statistics for each population in a database of reference populations.

Usage

```
genogeo(profile, df, CI = 0.95, min_n = 75, grouping = "pop",
  tilt = FALSE, ...)
```

Arguments

profile	The AIMs profile encoded as returned by the profile_AA_x0 function.
df	The database of reference populations as returned by the pops_to_DB function.
CI	The confidence level used to reject or accept the various hypotheses (between 0 and 1).
min_n	Minimum number of individuals in each database sample
grouping	should "pop" (the default) or "meta" be used for aggregating the results. Can also be "cluster" if this variable is defined in the input database.
tilt	Should exponential titling be used to obtain more accurate p-values in the distribution's tail (currently not implemented)
...	Further arguments that are passed to other functions

Value

A tibble containing the z-scores, p-values etc for each population.

Examples

```
df_ <- simulate_pops(pop_n = 20, aims_n = 50)
df_db <- pops_to_DB(df_)
profile <- random_AIMs_profile(df_db, keep_pop = TRUE)
profile$pop[1] # The true population
result <- genogeo(profile[,c("locus","x0")], df = df_db)
```

`genogeographer` *genogeographer: Methods for analysing forensic Ancestry Informative Markers*

Description

The genogeographer package provides: genogeo()

genogeo functions

See ?genogeo

`kidd_loci` *Kenn Kidd Lab markers*

Description

List of markers identified by Kenn Kidd lab.

Usage

```
kidd_loci
```

Format

List of 55 markers

locus Locus/Marker names

Source

K.K. Kidd et al.Progress toward an efficient panel of SNPs for ancestry inference. Forensic Science International: Genetics 10 (2014) 23–32

LR_table *Compute pairwise likelihood ratios*

Description

For each pair of a specified vector of profiles the likelihood ratios are computed. The list can include all populations in the data or only a subset. We may for inferral purposes restrict to ratios including at least one "accepted" population.

Usage

```
LR_table(result_df, lr_populations = NULL, only_accepted = TRUE,
  CI = 0.95, digits = NULL, keep_logP = FALSE)
```

Arguments

result_df	The output from genogeo
lr_populations	A vector of population names (pop in result_df). If NULL all populations are used.
only_accepted	Restrict the ratios to include minimum one accepted population.
CI	The level of confidence interval to be computed
digits	If rounding of the output should be performed.
keep_logP	Logical. Should the logP's be returned in output

Value

A tibble with numerator and denominator populations with their log10 LR and uncertainty.

Author(s)

Torben Tvedebrink <tvede@math.aau.dk>

Examples

```
df_ <- simulate_pops(pop_n = 4, aims_n = 50)
df_db <- pops_to_DB(df_)
profile <- random_AIMs_profile(df_db, keep_pop = TRUE)
profile$pop[1] # The true population
result <- genogeo(profile[,c("locus","x0")], df = df_db)
LR_table(result)
```

main_alleles *AIMs markers in Precision ID Ancestry Panel (Thermo Fisher Scientific)*

Description

List of markers with their main and alternative allele. The markers is the union of Seldin's and Kidd's markers.

Usage

```
main_alleles
```

Format

List of 164 markers

locus Locus/Marker names

main_allele The main allele (alleles are in lexicographic order)

other_allele The other variant

map_plot *Plot LTR z-scores on map*

Description

Plots the results from LRT on a map based on lat/lon info in the database. If no location is found in the data (e.g. using `simulte_pops`) nothing is plotted.

Usage

```
map_plot(data)
```

Arguments

`data`	The output from the `genogeo` function

Value

A map with population z-scores at their geographic origin

Author(s)

Torben Tvedebrink, <tvede@math.aau.dk>

Examples

```
df_ <- simulate_pops(pop_n = 4, aims_n = 50)
df_db <- pops_to_DB(df_)
profile <- random_AIMs_profile(df_db, keep_pop = TRUE)
profile$pop[1] # The true population
result <- genogeo(profile[,c("locus","x0")], df = df_db, min_n = 0)
result$lon <- runif(n = 4, min = -125, max = 125)
result$lat <- runif(n = 4, min = -50, max = 80)
## Not run: map_plot(result)
```

`pops_to_DB` *Pre-compute the scores for a given reference database*

Description

Convert the counts from each population over a range of AIMs SNPs q to observed likelihood ratio test, its mean and variance. Based on these pre-computed the evaluation of a specific profile is done using `genogeo` with the resulting dataframe as `df`.

Usage

```
pops_to_DB(db, ...)
```

Arguments

`db`	A dataframe with columns similar to those of `simulate_pops()`. If `db` contains information (recommended!) about "meta" (meta population) and "lat"/"lon" (location) these are carried over into the calculations
`...`	Additional arguments passed to `score_add_df`

Value

A tibble with population and locus specific score information

Examples

```
df_ <- simulate_pops(pop_n = 4, aims_n = 50)
df_db <- pops_to_DB(df_)
```

profile_AA_x0 *Function that compute the genotype probability for each population (rows in df)*

Description

Function that compute the genotype probability for each population (rows in df)

Usage

```
profile_AA_x0(AA_profile, df, select = c("locus", "x0"),
  keep_dropped = FALSE)
```

Arguments

AA_profile	A tibble/data.frame with columns 'locus', 'A1' and 'A2' holding the separated version of a genotype, eg. AG -> A1: A, A2: G
df	The database with main alleles per locus
select	Which columns to return
keep_dropped	Logical. Keep the non-matching alleles (compared to 'db') and those with genotype 'NN'

profile_admixture *Compute the z-score (and more) for admixed hypotheses*

Description

Compute the z-score (and more) for admixed hypotheses

Usage

```
profile_admixture(x0, df, hyp = NULL, grouping = "meta",
  return_all = FALSE, calc_logP = TRUE, ...)
```

Arguments

x0	A data frame/tibble with two columns: 'locus' and 'x0'
df	A tibble of reference profiles (as for 'genogeo')
hyp	If NULL all levels of 'grouping' is crossed and looped over as pairwise hypotheses. If a single level of 'grouping', this value is crossed with the remaining levels. If vector of two levels this is the only tested hypothesis.
grouping	Should the calculations be for meta populations ("meta") or sample populations ("pop")?
return_all	Should z-score be returned (FALSE) or all locus results (TRUE)?
calc_logP	Should log P(GenolHyp) be calculated (TRUE) or not (FALSE)?
...	additional arguments passed on to other functions

Value

A tibble of z-scores, or a list of pairwise results if 'return_all = TRUE'

`random_AIMs_profile` *Simulate a random AIMs profile*

Description

Use the information from `pops_to_DB` to simulate a profile from a random or given population. The sampling is done with respect to the null hypothesis, such that the total count is adjusted accordingly. For further details see Tvedebrink et al (2018), Section 3.1 (Simulations).

Usage

```
random_AIMs_profile(df, grouping = "pop", population = NULL,
  n = FALSE, keep_pop = FALSE)
```

Arguments

`df`	Database of reference profiles as returned by `pops_to_DB`
`grouping`	Simualte from `pop` (default) or `meta`.
`population`	The population to sample from. If NULL chosen at random.
`n`	Use numbers of samples as weights to choose the population randomly
`keep_pop`	Keep information on population

Author(s)

Torben Tvedebrink <tvede@math.aau.dk>

`seldin_loci` *Seldin Lab markers*

Description

List of markers identified by Seldin lab.

Usage

```
seldin_loci
```

Format

List of 122 markers

locus Locus/Marker names

Source

Kosoy et al. Ancestry Informative Marker Sets for Determining Continental Origin and Admixture Proportions in Common Populations in America. HUMAN MUTATION, Vol. 30, No. 1, 69–78, 2009.

`simulate_pops` *Simulate random populations*

Description

Simulate random populations

Usage

```
simulate_pops(pop_n = 100, pop_names = NULL, pop_totals = NULL,
  aims_n = 50, aims_names = NULL)
```

Arguments

`pop_n`	Number of populations to simulate
`pop_names`	Their names. If NULL: The names are "pop_001" through "pop_pop_n"
`pop_totals`	How many observations/sampled individuals per population. If one number this is used as parameter in a Poisson distribution
`aims_n`	Number of AIMs
`aims_names`	Their names. If NULL: The names are "rs_001" through "rs_aims_n"

Author(s)

Torben Tvedebrink <tvede@math.aau.dk>

Index