Package: admixr (via r-universe)

September 7, 2024

Title An Interface for Running 'ADMIXTOOLS' Analyses

Version 0.9.1.9000

Description An interface for performing all stages of 'ADMIXTOOLS' analyses (<https://reich.hms.harvard.edu/software>) entirely from R. Wrapper functions (D, f4, f3, etc.) completely automate the generation of intermediate configuration files, run 'ADMIXTOOLS' programs on the command-line, and parse output files to extract values of interest. This allows users to focus on the analysis itself instead of worrying about low-level technical details. A set of complementary functions for processing and filtering of data in the 'EIGENSTRAT' format is also provided.

License MIT + file LICENSE

URL https://github.com/bodkan/admixr

BugReports https://github.com/bodkan/admixr/issues

Depends R (>= 3.6.0)

Imports dplyr, magrittr, readr, stringr, tibble, stats, rlang, utils

Suggests glue, ggplot2, testthat, forcats, tidyr, knitr, rmarkdown

SystemRequirements ADMIXTOOLS suite of command-line utilities for population genetics. See <https://reich.hms.harvard.edu/software> for the most recent installation instructions and further information.

Encoding UTF-8

RoxygenNote 7.3.1

VignetteBuilder knitr

Repository https://bodkan.r-universe.dev

RemoteUrl https://github.com/bodkan/admixr

RemoteRef HEAD

RemoteSha 2a45a27a1dea6ed5fa3a6d2baf7a2ed587193d71

Contents

`count_snps` *Count the number/proportion of present/missing sites in each sample*

Description

Count the number/proportion of present/missing sites in each sample

Usage

```
count_snps(data, missing = FALSE, prop = FALSE)
```

Arguments

`data`	EIGENSTRAT data object.
`missing`	Count present SNPs or missing SNPs?
`prop`	Calculate the proportion instead of counts?

Value

A data.frame object with SNP counts/proportions.

Examples

```
## Not run: snps <- eigenstrat(download_data(dirname = tempdir()))

present_count <- count_snps(snps)
missing_count <- count_snps(snps, missing = TRUE)

present_proportion <- count_snps(snps, prop = TRUE)
missing_proportion <- count_snps(snps, missing = TRUE, prop = TRUE)

## End(Not run)
```

download_data *Download example SNP data.*

Description

The data is downloaded to a temporary directory by default.

Usage

```
download_data(dirname = tempdir())
```

Arguments

dirname	Directory in which to put the data (EIGENSTRAT trio of snp/geno/ind files).

eigenstrat *EIGENSTRAT data constructor*

Description

This function creates an instance of the EIGENSTRAT S3 class, which encapsulates all paths to data files required for an ADMIXTOOLS analysis.

Usage

```
eigenstrat(prefix = NULL, ind = NULL, snp = NULL, geno = NULL, exclude = NULL)
```

Arguments

prefix	Shared path to an EIGENSTRAT trio (set of ind/snp/geno files).
ind, snp, geno	Paths to individual EIGENSTRAT components.
exclude	Pre-defined snp file with excluded sites.

Value

S3 object of the EIGENSTRAT class.

Examples

```
## Not run: # download an example genomic data and get the path prefix to the
# trio of snp/geno/ind files in an EIGENSTRAT format
prefix <- download_data(dirname = tempdir())

# wrap the trio of snp/geno/ind files in an object of the class
# EIGENSTRAT
snps <- eigenstrat(prefix)

## End(Not run)
```

`f4ratio` *Calculate the D, f4, f4-ratio, or f3 statistic.*

Description

Calculate the D, f4, f4-ratio, or f3 statistic.

Usage

```
f4ratio(data, X, A, B, C, O, outdir = NULL, params = NULL)

d(
  data,
  W,
  X,
  Y,
  Z,
  quartets = NULL,
  outdir = NULL,
  f4mode = FALSE,
  params = NULL
)

f4(data, W, X, Y, Z, quartets = NULL, outdir = NULL, params = NULL)

f3(data, A, B, C, outdir = NULL, inbreed = FALSE, params = NULL)
```

Arguments

`data`	EIGENSTRAT data object.
`outdir`	Where to put all generated files (temporary directory by default).

`params` Named list of parameters and their values. For instance, `params = list(allsnps = "YES")` or `params = list(blgsize = 0.01)` (or an arbitrary combination of parameters using a list with multiple named elements).

`W, X, Y, Z, A, B, C, O`
Population names according to the nomenclature used in Patterson et al., 2012.

`quartets` List of character vectors (quartets of population/sample labels)

`f4mode` Calculate the f4 statistic instead of the D statistic.

`inbreed` See README.3PopTest in ADMIXTOOLS for an explanation.

Value

Data frame object with calculated statistics

Examples

```
## Not run: # download an example genomic data set and prepare it for analysis
snps <- eigenstrat(download_data(dirname = tempdir()))

# define a set of populations to analyze
pops <- c("French", "Sardinian", "Han", "Papuan", "Dinka")

result_f4ratio <- f4ratio(
    X = pops, A = "Altai", B = "Vindija", C = "Yoruba", O = "Chimp",
    data = snps
)

result_d <- d(
    W = pops, X = "Yoruba", Y = "Vindija", Z = "Chimp",
    data = snps
)

result_f4 <- f4(
    W = pops, X = "Yoruba", Y = "Vindija", Z = "Chimp",
    data = snps
)

result_f3 <- f3(
    A = pops, B = "Mbuti", C = "Khomani_San",
    data = snps
)

## End(Not run)
```

`filter_bed` *Filter EIGENSTRAT data based on a given BED file*

Description

Keep (or discard) SNPs that overlap (or lie outside of) regions in a given BED file.

Usage

```
filter_bed(
  data,
  bed,
  remove = FALSE,
  outfile = tempfile(fileext = ".snp"),
  bedtools_args = ""
)
```

Arguments

`data`	EIGENSTRAT data object.
`bed`	Path to a BED file.
`remove`	Remove sites falling inside the BED file regions? By default, sites that do not overlap BED regions are removed.
`outfile`	Path to an output snp file with coordinates of excluded sites.
`bedtools_args`	Optional arguments to ‘bedtools intersect‘ such as `"-sorted"` or `"-sorted -nonamecheck"`.

Details

This function requires a functioning bedtools installation! See:

- https://github.com/arq5x/bedtools2

- https://bedtools.readthedocs.io/

Value

Updated S3 EIGENSTRAT data object.

Examples

```
## Not run: # download an example genomic data set
prefix <- download_data(dirname = tempdir())
# create an EIGENSTRAT R object from the downloaded data
snps <- eigenstrat(prefix)

# get the path to an example BED file
bed <- file.path(dirname(prefix), "regions.bed")

# BED file contains regions to keep in an analysis
snps_kept <- filter_bed(snps, bed)
# BED file contains regions to remove from an analysis
snps_removed <- filter_bed(snps, bed, remove = TRUE)
```

```
## End(Not run)
```

loginfo *Print the full log output of an admixr wrapper to the console.*

Description

Print the full log output of an admixr wrapper to the console.

Usage

```
loginfo(x, target = NA, save = FALSE, prefix = NA, dir = ".", suffix = ".txt")
```

Arguments

x	Output from one of the admixr wrappers (d, f4, qpAdm, ...)
target	A specific log to examine (relevant for multiple target qpAdm runs)
save	Save the log output to a disk?
prefix	Prefix of the output log file(s) (name of the admixr command by default)
dir	In which directory to save the log file(s)?
suffix	Suffix of the output log file(s) (".txt" by default)

Examples

```
## Not run: # download an example genomic data set and prepare it for analysis
snps <- eigenstrat(download_data(dirname = tempdir()))

# define a set of populations to analyze and calculate a D statistic
pops <- c("French", "Sardinian", "Han", "Papuan", "Khomani_San", "Mbuti", "Dinka")
result_d <- d(
    W = pops, X = "Yoruba", Y = "Vindija", Z = "Chimp",
    data = snps
)

# examine the full log output associated with the returned object
loginfo(result_d)

## End(Not run)
```

merge_eigenstrat *Merge two sets of EIGENSTRAT datasets*

Description

This function utilizes the 'mergeit' command distributed in ADMIXTOOLS.

Usage

```
merge_eigenstrat(merged, a, b, strandcheck = "NO")
```

Arguments

merged	Prefix of the path to the merged EIGENSTRAT snp/ind/geno trio.
a, b	Two EIGENSTRAT objects to merge.
strandcheck	Deal with potential strand issues? Mostly for historic reasons. For details see the README of ADMIXTOOLS convertf.

print.admixr_result *Print out the admixr result object (dataframe or a list) without showing the hidden attributes.*

Description

Print out the admixr result object (dataframe or a list) without showing the hidden attributes.

Usage

```
## S3 method for class 'admixr_result'
print(x, ...)
```

Arguments

x	admixr output object (dataframe or a list produced by qpAdm/qpWave)
...	Additional arguments passed to print.

`print.EIGENSTRAT` *EIGENSTRAT print method*

Description

Print EIGENSTRAT object components.

Usage

```
## S3 method for class 'EIGENSTRAT'
print(x, ...)
```

Arguments

`x` EIGENSTRAT data object.

`...` Further arguments passed to or from other methods.

`qpAdm` *Calculate ancestry proportions in a set of target populations.*

Description

Calculate ancestry proportions in a set of target populations.

Usage

```
qpAdm(
  data,
  target,
  sources,
  outgroups,
  outdir = NULL,
  params = list(allsnps = "YES", summary = "YES", details = "YES")
)
```

Arguments

`data` EIGENSTRAT data object.

`target` Vector of target populations (evaluated one at a time).

`sources` Source populations related to true ancestors.

`outgroups` Outgroup populations.

`outdir` Where to put all generated files (temporary directory by default).

`params` Named list of parameters and their values. For instance, `params = list(allsnps = "YES")` or `params = list(blgsize = 0.01)` (or an arbitrary combination of parameters using a list with multiple named elements).

Value

List of three components: 1. estimated ancestry proportions 2. ranks statistics 3. analysis of patterns (all possible subsets of ancestry sources).

Examples

```
## Not run: # download example data set and prepare it for analysis
snps <- eigenstrat(download_data(dirname = tempdir()))

# estimate the proportion of Neandertal ancestry in a French
# individual and other associated qpAdm statistics (see detailed
# description in the tutorial vignette)
result <- qpAdm(
    target = "French",
    sources = c("Vindija", "Yoruba"),
    outgroups = c("Chimp", "Denisova", "Altai"),
    data = snps
)

## End(Not run)
```

qpAdm_filter *Filter qpAdm rotation results for only 'sensible' models*

Description

Filter for p-value larger than a specified cuttof and admixture proportions between 0 and 1.

Usage

```
qpAdm_filter(x, p = 0.05)
```

Arguments

x	Output of a qpAdm_rotation() function
p	p-value cutoff (default 0: will only filter for sensible admixture proportions)

Value

qpAdm_rotation object filtered down based on p-value

Examples

```
## Not run: # download an example genomic data set and prepare it for analysis
snps <- eigenstrat(download_data(dirname = tempdir()))

# find the set of most likely two-source qpAdm models of
# a French individual - produce only the 'proportions'
# qpAdm summary
models <- qpAdm_rotation(
    data = snps,
    target = "French",
    candidates = c("Dinka", "Mbuti", "Yoruba", "Vindija",
                   "Altai", "Denisova", "Chimp"),
    minimize = TRUE,
    nsources = 2,
    ncores = 2,
    fulloutput = FALSE
)

# filter out models which can clearly be rejected
fits <- qpAdm_filter(models, p = 0.05)

## End(Not run)
```

---

qpAdm_rotation *Fit qpAdm models based on the rotation strategy described in Harney et al. 2020 (bioRxiv)*

Description

Fit qpAdm models based on the rotation strategy described in Harney et al. 2020 (bioRxiv)

Usage

```
qpAdm_rotation(
  data,
  target,
  candidates,
  minimize = TRUE,
  nsources = 2,
  ncores = 1,
  fulloutput = FALSE,
  params = NULL
)
```

Arguments

`data`	EIGENSTRAT dataset

`target` Target population that is modeled as admixed

`candidates` Potential candidates for sources and outgroups

`minimize` Test also all possible subsets of outgroups? (default TRUE)

`nsources` Number of sources to pull from the candidates

`ncores` Number of CPU cores to utilize for model fitting

`fulloutput` Report also ’ranks’ and ’subsets’ analysis from qpAdm in addition to the admixture proportions results? (default FALSE)

`params` Named list of parameters and their values to be passed to `qpAdm()`.

Value

qpAdm list with proportions, ranks and subsets elements (as with a traditional qpAdm run) or just the proportions (determined by the value of the ’fulloutput’ argument)

Examples

```
## Not run: # download an example genomic data set and prepare it for analysis
snps <- eigenstrat(download_data(dirname = tempdir()))

# find the set of most likely two-source qpAdm models of
# a French individual - produce only the 'proportions'
# qpAdm summary
models <- qpAdm_rotation(
    data = snps,
    target = "French",
    candidates = c("Dinka", "Mbuti", "Yoruba", "Vindija",
                   "Altai", "Denisova", "Chimp"),
    minimize = TRUE,
    nsources = 2,
    ncores = 2,
    fulloutput = FALSE
)

## End(Not run)
```

qpWave *Find the most likely number of ancestry waves using the qpWave method.*

Description

Given a set of ’left’ populations, estimate the lowest number of necessary admixture sources related to the set of ’right’ populations.

Usage

```
qpWave(
  data,
  left,
  right,
  maxrank = NULL,
  details = FALSE,
  outdir = NULL,
  params = NULL
)
```

Arguments

`data` EIGENSTRAT data object.

`left, right` Character vectors of populations labels.

`maxrank` Maximum rank to test for.

`details` Return the A, B matrices used in rank calculations?

`outdir` Where to put all generated files (temporary directory by default).

`params` Named list of parameters and their values. For instance, `params = list(allsnps = "YES")` or `params = list(blgsize = 0.01)` (or an arbitrary combination of parameters using a list with multiple named elements).

Details

It has been shown (Reich, Nature 2012 - Reconstructing Native American population history) that if the 'left' populations are mixtures of N different sources related to the set of 'right' populations, the rank of the matrix of the form $f_4(left_i, left_j; right_k, right_l)$ will have a rank N - 1. This function uses the ADMIXTOOLS command qpWave to find the lowest possible rank of this matrix that is consistent with the data.

Value

Table of rank test results.

Examples

```
## Not run: # download example data set and prepare it for analysis
snps <- eigenstrat(download_data(dirname = tempdir()))

# run the qpWave wrapper (detailed description in the tutorial vignette)
result <- qpWave(
     left = c("French", "Sardinian", "Han"),
     right = c("Altai", "Yoruba", "Mbuti"),
     data = snps
)

## End(Not run)
```

`read_ind` *Read an EIGENSTRAT ind/snp/geno file.*

Description

These functions each read one part of the EIGENSTRAT dataset trio.

Usage

```
read_ind(data)

read_snp(data, exclude = FALSE)

read_geno(data)
```

Arguments

`data`	EIGENSTRAT data object.
`exclude`	Read the list of excluded SNPs?

Details

Note that `read_geno()` will only read plain-text geno files, not compressed ones.

Value

A data.frame object.

`read_output` *Read an output file from one of the ADMIXTOOLS programs.*

Description

Read an output file from one of the ADMIXTOOLS programs.

Usage

```
read_output(file, ...)
```

Arguments

`file`	A path to an output file.
`...`	See the 'details' argument of qpWave.

Value

A tibble with the results.

relabel *Change labels of populations or samples*

Description

Replace population/sample names with specified group labels.

Usage

```
relabel(data, ..., outfile = tempfile(fileext = ".ind"))
```

Arguments

`data`	EIGENSTRAT trio.
`...`	Population/sample names to merge (each new group defined as a character vector).
`outfile`	Path to an output snp file with coordinates of excluded sites.

Value

Updated S3 EIGENSTRAT data object with an additional 'group' slot specifying the path to a new ind file. #'

Examples

```
## Not run: # download an example genomic data set and prepare it for analysis
snps <- eigenstrat(download_data(dirname = tempdir()))

# group individual samples into larger populations, creating a new
# EIGENSTRAT R object
new_snps <- relabel(
    snps,
    European = c("French", "Sardinian"),
    African = c("Dinka", "Yoruba", "Mbuti", "Khomani_San"),
    Archaic = c("Vindija", "Altai", "Denisova")
)

## End(Not run)
```

reset *Reset modifications to an EIGENSTRAT object*

Description

Set 'exclude' and 'group' modifications of snp and ind files, effectively resetting the dataset into its original state.

Usage

```
reset(data)
```

Arguments

`data`	EIGENSTRAT data object.

Value

EIGENSTRAT data S3 object.

Examples

```
## Not run: # download an example genomic data set and prepare it for analysis
snps <- eigenstrat(download_data(dirname = tempdir()))

# group individual samples into larger populations, creating a new
# EIGENSTRAT R object
new_snps <- relabel(
    snps,
    European = c("French", "Sardinian"),
    African = c("Dinka", "Yoruba", "Mbuti", "Khomani_San"),
    Archaic = c("Vindija", "Altai", "Denisova")
)

# remove the population grouping in the previous step - this
# results in the same EIGENSTRAT object tht we started with
original_snps <- reset(new_snps)

## End(Not run)
```

transversions_only *Remove transversions (C->T and G->A substitutions)*

Description

Remove substitutions that are more likely to be a result of ancient DNA damage (C->T and G->A substitutions).

Usage

```
transversions_only(data, outfile = tempfile(fileext = ".snp"))
```

Arguments

`data`	EIGENSTRAT data object.
`outfile`	Path to an output snp file with coordinates of excluded sites.

Value

Updated S3 EIGENSTRAT data object with an additional 'exclude' slot specifying the path to the set of SNPs to be removed from a downstream analysis.

Examples

```
## Not run: # download an example genomic data set and prepare it for analysis
snps <- eigenstrat(download_data(dirname = tempdir()))

# perform the calculation only on transversions
snps_tv <- transversions_only(snps)
results_d <- d(W = "French", X = "Dinka", Y = "Altai", Z = "Chimp", data = snps_tv)

## End(Not run)
```

write_ind *Write an EIGENSTRAT ind/snp/geno file.*

Description

Write an EIGENSTRAT ind/snp/geno file.

Usage

```
write_ind(df, file)

write_snp(df, file)

write_geno(df, file)
```

Arguments

`df` A data.frame object.

`file` Path to an output file.

Index