Package ‘gstudio’

February 20, 2015

Maintainer Rodney J. Dyer <rjdyer@vcu.edu>
License GPL (>= 2)
Title Analyses and functions related to the spatial analysis of genetic marker data.
Type Package
Author Rodney J. Dyer <rjdyer@vcu.edu>
Description This package contains classes and functions for the spatial analysis of marker data to include structure statistics, analyses based upon allele frequencies.
Version 1.3
Date 2014-04-01
LazyLoad true
VignetteBuilder knitr
Depends R (>= 3.0), ggplot2, raster
Imports sp, grid, reshape2, ggmap, popgraph
Suggests testthat, knitr
NeedsCompilation no
Repository CRAN
Date/Publication 2014-06-03 01:50:40

R topics documented:

  gstudio-package ........................................... 3
  +.locus .................................................. 5
  -.locus .................................................. 6
  A .......................................................... 7
  Ae ........................................................ 7
  alleles ..................................................... 8
  arapat ..................................................... 9
  as.data.frame.locus ...................................... 9
  as.list.locus ........................................... 10
R topics documented:

as.locus ................................................................. 10
bad.parents ......................................................... 11
c.locus ................................................................. 12
column_class ......................................................... 12
cornus_florida ...................................................... 13
data_2_column ...................................................... 13
data_aflp ............................................................... 14
data_separated ....................................................... 14
data.snp ................................................................. 14
data_zymelike ......................................................... 14
Dest ................................................................... 15
dist_amova ............................................................ 16
dist_bray ............................................................... 17
dist_cavalli ............................................................ 18
dist_egd ............................................................... 19
dist_euclidean ....................................................... 19
dist_jaccard .......................................................... 20
dist_nei ............................................................... 21
dist_ss ................................................................. 22
exclusion_probability ........................................... 22
Fis .................................................................... 23
frequencies ......................................................... 24
genetic_distance ..................................................... 25
genetic_diversity .................................................... 26
ngenetic_relatedness ............................................. 27
genetic_structure ................................................... 28
genotype_frequencies ............................................. 29
geom_frequencies ................................................ 30
geom_locus .......................................................... 31
geom_strata ........................................................ 32
geom_surface ...................................................... 32
grid.csv ............................................................. 33
Gst ................................................................. 33
Gst_prime ........................................................... 34
He ................................................................... 35
Ho ................................................................. 36
indices ............................................................. 36
is.locus ............................................................. 37
is.na.locus ........................................................ 37
is_frequency ........................................................ 38
is_heterozygote .................................................... 38
kronecker_delta .................................................... 39
locus ............................................................... 40
make_population .................................................. 41
mate ............................................................... 42
migrate ............................................................ 43
minus_mom .......................................................... 44
multilocus_diversity ............................................. 45
gstudio is a suite of tools used in the spatial analysis of population genetic data created for the book "Applied Landscape Genetics" by R.J. Dyer. This version of the package (1.X) is a significant de-viant from the previous (versions 0.X) resulting from an entire re-write of the package to maximize ease of use and algorithm quickness. As such, previous scripts using gstudio (vers. 0.X) will need to be refactored a bit. While this can be a bit of a pain, the larger utility of the new format will pay off in spades.
Details
There are some very good examples of the components of this package are used in the vignettes for this package.

Author(s)

Rodney J. Dyer <rjdyer@vcu.edu>

Description

An overload of the + operator for locus objects that results in the creation of an offspring locus.

Usage

```r
## S3 method for class 'locus'
e1 + e2
```

Arguments

- `e1`: A locus object.
- `e2`: A locus object.

Value

A new locus object that represents an offspring genotype.

Author(s)

Rodney J. Dyer <rjdyer@vcu.edu>

Examples

```r
dad <- locus( c("A", "A") )
mom <- locus( c("B", "B") )
off <- mom + dad
off
```
-locus

Overload '-' operator for pairs of locus objects

Description

An overload of the - operator for locus objects that removes the contribution of a parental genotype to an offspring genotype (if possible).

Usage

```r
## S3 method for class 'locus'
e1 - e2
```

Arguments

- `e1`: A locus object representing the offspring.
- `e2`: A locus object representing the parent.

Value

A new locus object that represents the genotypes left over after removing the parental part (if possible).

Note

In some cases it is not possible to remove parental alleles (e.g., consider cases where both parent and offspring are the same heterozygote).

Author(s)

Rodney J. Dyer <rjdyer@vcu.edu>

Examples

```r
mom <- locus( c("A", "A") )
off <- locus( c("A", "B") )
dad.gamete <- off - mom
dad.gamete
```
Allelic Diversity

Description
This function returns the number of unique, non-NA alleles, in a sample.

Usage
A(x, min_freq = 0)

Arguments
x A set of locus objects
min_freq The minimum frequency of allele to consider (default=0)

Value
The number of alleles that have min_freq frequency.

Examples
loci <- c( locus(1:2), locus(c(1,1)), locus(c(2,2)), locus(2:3) )
A(loci)
A(loci, min_freq=0.13)

Effective Allelic Diversity

Description
This function returns the effective number of alleles in a sample.

Usage
Ae(x)

Arguments
x A set of locus objects either as a vector or within a data.frame.

Value
The effective number of alleles either as a numeric value or as a data.frame if multiple loci are passed.
Examples

```r
locus <- c( locus(1:2), locus(c(1,1)), locus(c(2,2)), locus(2:3) )
Ae(locus)
locus2 <- c( locus(1:2), locus(c(1,1)), locus(c(2,2)), locus(2:3) )
df <- data.frame( locus, locus2 )
Ae(df)
```

### alleles

*Returns the alleles in a locus object*

Description

This function will provide the alleles within a locus object for either a single locus or for a vector of loci.

Usage

```r
alleles(x, all = TRUE)
```

```r
## Default S3 method:
alleles(x, all = TRUE)
```

```r
## S3 method for class 'locus'
alleles(x, all = TRUE)
```

Arguments

- `x`: A locus object (single or vector)
- `all`: A flag indicating that you want all alleles returned (default=TRUE) A FALSE flag here indicates that you only want the unique alleles returned (e.g., the ones present).

Value

A matrix of alleles. If `x` is a vector then the result will be represented by rows.

Names from passed vector

Either a matrix or vector of alleles depending upon what was passed to the function.

Author(s)

Rodney J. Dyer <rjdyer@vcu.edu>

Examples

```r
locus <- c( locus(1:2), locus(c(1,1)), locus(c(2,2)), locus(2:3) )
alleles(locus)
```
Data for Araptus attenuatus

Description

This is an EPIC genetic dataset for the Sonoran desert beetle Araptus attenuatus.

as.data.frame.locus Convert locus to a data frame

Description

This converts a locus object into a data.frame so that you can use locus objects in a column of a data.frame.

Usage

## S3 method for class 'locus'
as.data.frame(x, ...)

Arguments

x 
An object of type code. This can be either a single genotype (a rare case) or a vector of genotypes (preferred).

... 
Additional objects that are passed to as.data.frame.vector.

Value

A data.frame object.

Note

If you do not assign a data name to the x in assignment (e.g., TPI=x) it will name the column in the data.frame the same as the name of the variable you assigned it. If this is confusing see the examples.

Author(s)

Rodney J. Dyer <rjdyer@vcu.edu>

Examples

loc1 <- c( locus(1:2), locus(1:2), locus( c(1,1) ) )
df <- data.frame( ID=1:3, NAMED_LOCUS=loc1, loc1 )
summary(df)
as.list.locus  
*Converts locus to a list*

**Description**

This converts a locus object into a list so that you can use locus objects in a column of a list.

**Usage**

```r
## S3 method for class 'locus'
as.list(x, ...)
```

**Arguments**

- `x`  
  An object of type `locus`. This can be either a single genotype (a rare case) or a vector of genotypes (preferred).

- `...`  
  Additional objects that are passed to `as.data.frame.vector`.

**Value**

A list object.

**Author(s)**

Rodney J. Dyer <rjdyer@vcu.edu>

**Examples**

```r
loc1 <- c(locus(1:2), locus(1:2), locus(c(1,1)))
df <- as.list(ID=1:3, NAMED_LOCUS=loc1, loc1)
df
```

---

as.locus  
*As operator for locus*

**Description**

This takes several things and shoves it into the constructor.

**Usage**

```r
as.locus(x)
```

**Arguments**

- `x`  
  An object that is to be turned into a locus.
bad_parents

Value

An object of type locus

Author(s)

Rodney J. Dyer <rjdyer@vcu.edu>

See Also

locus

Examples

lst <- list( "A", "B" )
as.locus( lst )
vec <- 1:2
as.locus( vec )
chr <- "A"
as.locus( chr )
chr.sep <- "A:A"
as.locus( chr )

bad_parents

Determines of the indicated parent can possibly be the parent of the offspring

Description

This function takes a data.frame full of parents and offspring and checks to see if the offspring are actually assigned to a compatible adult.

Usage

bad_parents(df, AdultID = "ID", OffID = "OffID", verbose = FALSE)

Arguments

df A data.frame with identification and locus columns.
AdultID The column header for id of the adults. All adults have unique identification numbers and all offspring have the the same number as the maternal/paternal individual from which they were sampled or are being compared. Default = "ID"
OffID A column indicating offspring identification numbers. By default all adults have OffID=0 (this is how I tell if they are really adults and not offspring), and all offspring have OffID != 0. Default = "OffID"
verbose Print out mismatched parent/offspring pairs.
Value

The data.frame with a new column, Is.Parent with values of TRUE/FALSE/NA (the NA is for adults).

---

c.locus | Concatinate locus objects

Description

An overload of the c function for locus objects.

Usage

```r
## S3 method for class 'locus'
c(..., recursive = FALSE)
```

Arguments

- `...`: The locus objects to be concatenated.
- `recursive`: A flag to do the concatenation recursively.

Value

A vector of locus objects

Author(s)

Rodney J. Dyer <rjdyer@vcu.edu>

Examples

```r
loci <- c( locus(1:2), locus(1:2), locus( c(1,1) ) )
loci
```

---

column_class | Find columns of data in a data.frame for a type of class

Description

This convenience function finds the data of a specified type in the passed data.frame.

Usage

```r
column_class(x, class, mode = c("label", "index")[1])
```
Arguments

x        An object of data.frame type.
class    The type of class to search for. If this is omitted then the classes of each column will be returned.
mode     How you would like the column references to be used. Current values are:
label    The name of the column label with the specified class.
index    The numerical index of the column with specified class.

Value

A list of labels or indices indicating where columns of the queried type exist or a list of all column classes.

Author(s)

Rodney J. Dyer <rjdyer@vcu.edu>

Examples

locus <- c( locus(1:2), locus(c(1,1)), locus(c(2,2)), locus(2:3) )
pop <- factor( c(rep("A",2), rep("B",2)))
df <- data.frame( Population=pop, X=runif(4), TPI=locus )
df
column_class(df,"factor")
column_class(df,"numeric")
column_class(df, "locus")
column_class(df,"locus",mode="index")

cornus_florida  Cornus mom/offspring data set

description

Example data set for demonstrating parent/offspring data

data_2_column  Two Column Example Data

description

Example data for 2-column data types
<table>
<thead>
<tr>
<th>data_type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>data_aflp</td>
<td><em>AFLP example data set</em></td>
</tr>
<tr>
<td>data_separated</td>
<td><em>Separated Example Data</em></td>
</tr>
<tr>
<td>data_snp</td>
<td><em>SNP Example Data</em></td>
</tr>
<tr>
<td>data_zymelike</td>
<td><em>Zyme-like Example Data</em></td>
</tr>
</tbody>
</table>

**Description**

This is an example of the AFLP data input types.

Example data for separated data types

Example data for SNP data types

Example data like zyme genotypes
### Estimation Joost’s Dest parameter

#### Description

This function estimates the parameter (and potentially the confidence surrounding its value) for Joost’s Dest.

#### Usage

```r
Dest(x, stratum = "Population", nperm = 0, size.correct = FALSE)
```

#### Arguments

- `x`  
  A vector of `locus` objects or a `data.frame` with `locus` objects.
- `stratum`  
  Either a vector of strata variables if `x` is a `locus` vector or the name of the column representing strata in `x` if it is a `data.frame`.
- `nperm`  
  The number of permutations to run for significance of the estimator.
- `size.correct`  
  A flag indicating that the estimate should be corrected for based upon sample sizes (default=TRUE).

#### Value

A `data.frame` with Dest, Hs, Ht, and P (if asked for). When multiple loci are provided, the results also provide a multilocus estimate using the harmonic mean.

#### Author(s)

Rodney J. Dyer <rjdyer@vcu.edu>

#### Examples

```r
a1 <- sample( LETTERS[1:5], size=20, replace=TRUE)
a2 <- sample( LETTERS[4:8], size=20, replace=TRUE)
raw_alleles <- matrix( c(a1,a2), ncol=2, byrow=TRUE )
locus <- locus( raw_alleles, type="column")
Dest( locus, Population )
a1 <- sample( LETTERS[1:5], size=20, replace=TRUE)
a2 <- sample( LETTERS[4:8], size=20, replace=TRUE)
raw_alleles <- matrix( c(a1,a2), ncol=2, byrow=TRUE )
locus2 <- locus( raw_alleles, type="column")
df <- data.frame( Population, TPI=locus, PGM=locus2 )
Dest( df, nperm=99)
```
dist_amova

Estimation of amova distance

Description

This function returns a measure of genetic distance based upon the AMOVA distance metric.

Usage

dist_amova(x)

Arguments

x  A dataframe with locus columns in it.

Value

The AMOVA distance matrix

Author(s)

Rodney J. Dyer <rjdyer@vcu.edu>

Examples

AA <- locus( c("A","A") )
AB <- locus( c("A","B") )
BB <- locus( c("B","B") )
AC <- locus( c("A","C") )
AD <- locus( c("A","D") )
BC <- locus( c("B","C") )
BD <- locus( c("B","D") )
CC <- locus( c("C","C") )
CD <- locus( c("C","D") )
DD <- locus( c("D","D") )
loci <- c(AA,AB,AC,AD,BB,BC,BD,CC,CD,DD)
D <- dist_amova( loci )
rownames(D) <- colnames(D) <- as.character(loci)
D
Description

This function returns a measure of genetic distance based upon the AMOVA distance metric.

Usage

```r
dist_bray(x, stratum = "Population")
```

Arguments

- `x`: Either a `data.frame` with both stratum and locus objects in them (for strata distance) OR a vector of locus objects and this will calculate distance based upon individual genetic distances.
- `stratum`: The name of the stratum variable in `x`

Value

A matrix of Jaccard distance

Author(s)

Rodney J. Dyer <rjdyer@vcu.edu>

Examples

```r
AA <- locus( c(“A”,”A”) )
AB <- locus( c(“A”,”B”) )
AC <- locus( c(“A”,”C”) )
BB <- locus( c(“B”,”B”) )
BC <- locus( c(“B”,”C”) )
CC <- locus( c(“C”,”C”) )
loci <- c(AA,AA,AB,AA,BB,BC,CC,BB,BB,CC)
df <- data.frame( Population=c(rep(“A”,5),rep(“B”,5) ), TPI=loci )
D <- dist_bray(df)
```
dist_cavalli

Estimation of Bray-Curtis distance

Description

This function returns a measure of genetic distance based upon the Cavalli-Sforza & Edwards (1967) distance metric. Assuming drift is the only source or differences observed among strata.

Usage

dist_cavalli(x, stratum = "Population")

Arguments

stratum The groups among which you are going to estimate genetic distances.
x The genetic data, either as a single locus or multilocus (data.frame) object.

Value

A matrix of Cavalli-Sforza Genetic distance estimates.

Author(s)

Rodney J. Dyer <rjdyer@vcu.edu>

Examples

AA <- locus( c("A","A") )
AB <- locus( c("A","B") )
BB <- locus( c("B","B") )
AC <- locus( c("A","C") )
AD <- locus( c("A","D") )
BC <- locus( c("B","C") )
BD <- locus( c("B","D") )
CC <- locus( c("C","C") )
CD <- locus( c("C","D") )
DD <- locus( c("D","D") )
loc1 <- c(AA,AB,AC,AD,BB,BC,BD,CC,CD,DD)
loc2 <- c(AA,AA,AC,AA,CC,CC,CC,CC,AA,AC)
df <- data.frame( Population=c(rep("Pop-A",5),rep("Pop-B",5)), TPI=loc1, PGM=loc2 )
**dist_cgd**

_Estimation of conditional genetic distance_

**Description**

This function returns a measure of conditional genetic distance based upon the Population Graphs approach from Dyer & Nason (2004) and Dyer et al. (2010).

**Usage**

```
dist_cgd(x, stratum = "Population")
```

**Arguments**

- `stratum` The groups among which you are going to estimate genetic distances.
- `x` The genetic data, either as a single locus or multilocus (data.frame) object.

**Value**

A matrix of conditional genetic distance estimates.

**Author(s)**

Rodney J. Dyer <rjdyer@vcu.edu>

---

**dist_euclidean**

_Estimation of euclidean distance_

**Description**

This function returns a measure of genetic distance based upon the euclidean frequency distance metric.

**Usage**

```
dist_euclidean(x, stratum = "Population")
```

**Arguments**

- `x` The genetic data, either as a single locus or multilocus (data.frame) object.
- `stratum` The groups among which you are going to estimate genetic distances.

**Value**

A matrix of euclidean distance estimates.
Author(s)

Rodney J. Dyer <rjdyer@vcu.edu>

Examples

```r
AA <- locus( c("A","A" )
AB <- locus( c("A","B" )
BB <- locus( c("B","B" )
loci <- c(AA,AA,AB,AA,BB,BB,AB,BB,AB)
df <- data.frame( Population=c(rep("A",5),rep("B",5) ), TPI=loci )
dist_jaccard(df)
```

Description

This function returns a measure of genetic distance based upon the Jaccard set distance metric.

Usage

```r
dist_jaccard(x, stratum = "Population")
```

Arguments

- **x**: A data.frame with both stratum and locus objects in them.
- **stratum**: The name of the stratum variable in x

Value

A matrix of Jaccard distance

Author(s)

Rodney J. Dyer <rjdyer@vcu.edu>

Examples

```r
AA <- locus( c("A","A" ))
AB <- locus( c("A","B" )
AC <- locus( c("A","C" )
BB <- locus( c("B","B" )
BC <- locus( c("B","C" )
CC <- locus( c("C","C" )
loci <- c(AA,AA,AB,AA,BB,BC,CC,BB,BB,CC)
df <- data.frame( Population=c(rep("A",5),rep("B",5) ), TPI=loci )
dist_jaccard(df)
```
Estimation of Bray-Curtis distance

Description

This function returns a measure of genetic distance based upon the Nei’s Genetic distance metric. Assumes drift/mutation equilibrium is the main force governing your observed differences.

Usage

```r
dist_nei(x, stratum = "Population")
```

Arguments

- `x`: A data frame with locus objects in it as well as a stratum object.
- `stratum`: A factor indicating the stratum to use.

Value

The Nei Genetic distance

Note

This is the bias corrected Nei’s Standard genetic distance.

Author(s)

Rodney J. Dyer <rjdyer@vcu.edu>

Examples

```r
AA <- locus( c("A","A") )
AB <- locus( c("A","B") )
BB <- locus( c("B","B") )
AC <- locus( c("A","C") )
AD <- locus( c("A","D") )
BC <- locus( c("B","C") )
BD <- locus( c("B","D") )
CC <- locus( c("C","C") )
CD <- locus( c("C","D") )
DD <- locus( c("D","D") )
loc1 <- c(AA,AB,AC,AD,BC,BD,CC,CD,DD)
loc2 <- c(AA,AA,AC,CC,AC,CC,AA,AC)
df <- data.frame( Population=c(rep("Pop-A",5),rep("Pop-B",5)), TPI=loc1, PGM=loc2 )
dist_nei( df )
```
### dist_ss  

*Estimation of SS distance*

**Description**

This function returns a measure of genetic distance based upon the partitioned Sums of Squares from the AMOVA analysis. There are no particular assumptions to this estimation.

**Usage**

```r
dist_ss(x, stratum = "Population")
```

**Arguments**

- `x`: A data frame with locus objects in it as well as a stratum object.
- `stratum`: A factor indicating the stratum to use.

**Value**

The SS distance matrix

**Author(s)**

Rodney J. Dyer <rjdyer@vcu.edu>

---

### exclusion_probability

*Paternity exclusion probability*

**Description**

This function estimates the multilocus exclusion probability from a set of loci.

**Usage**

```r
exclusion_probability(x)
```

**Arguments**

- `x`: A data.frame returned from `frequencies()`

**Value**

A data.frame with single and potentially multilocus exclusion probabilities

**Author(s)**

Rodney J. Dyer <rjdyer@vcu.edu>
Examples

```r
loci <- c(locus(1:2), locus(c(1,1)), locus(c(2,2)))
freqs <- frequencies( loci )
exclusion_probability( freqs )
```

---

### Fis

*Estimate simple inbreeding from frequencies*

#### Description

Returns the general Fis = 1-ho/he parameter from the locus being passed or a set of them if the value passed is a `data.frame` with locus objects.

#### Usage

```r
Fis(x, small.sample.correction = FALSE)
```

#### Arguments

- `x` Either a locus object or a `data.frame` with locus objects.
- `small.sample.correction` Passes this along to `He` for small sample sizes.

#### Value

The inbreeding F statistic as a numeric value or a `data.frame` if you passed multiple loci to this function.

#### Author(s)

Rodney J. Dyer <rjdyer@vcu.edu>

#### Examples

```r
loci <- c( locus(c("A","A") ), locus(c("A","A") ), locus(c("A","B") ) )
Fis( loci )
```
frequencies

Get allele frequencies

Description

This function converts loci into data frames consisting of alleles, stratum, frequencies, etc.

Usage

frequencies(x, loci, stratum, ...)

## Default S3 method:
frequencies(x, ...)

## S3 method for class 'locus'
frequencies(x, loci, ...)

## S3 method for class 'data.frame'
frequencies(x, loci, stratum, ...)

Arguments

x        Either a vector of types locus or a data.frame containing locus objects.
loci     The loci to get frequencies from (default is all).
stratum  Partition of the data based upon this stratum (default is no partitions).
...      Ignored

Value

A data frame with Frequencies, Alleles, Loci, and perhaps Stratum columns (Allele and Frequencies are at a minimum).
A data frame with Frequencies, Alleles, Loci, and perhaps Stratum columns (Allele and Frequencies are at a minimum).
A data frame with Frequencies, Alleles, Loci, and perhaps Stratum columns (Allele and Frequencies are at a minimum).
A data frame with Frequencies, Alleles, Loci, and perhaps Stratum columns (Allele and Frequencies are at a minimum).

Note

You can submit RAD-seq genotypes in a normal data.frame to this function BUT it must be formatted as follows. Each locus is represented by three columns of probabilities (must sum to 1). You must label the columns of your data frame with the name of the locus and a column number separated by an underscore '_' (e.g., Loc1_1, Loc1_2, Loc1_3, Loc2_1, Loc2_2, Loc2_3, etc). The function will name the locus the part before the dash (e.g., Loc1). The frequencies of the heterozygote genotype frequency represented in the middle position.
**genetic_distance**

*Estimate genetic distances among strata in a.*

**Description**

This function is the main one used for estimating genetic distances among either individuals or stratum. Given the large number of genetic distance metrics, some are recreated here, de novo, and some are estimated through other existing R packages.

**Usage**

```r
genetic_distance(x, stratum = "Population", mode)
```

**Arguments**

- `x` A data.frame object with `locus` columns.
- `stratum` The strata by which the genetic distances are estimated. This can be an optional parameter when estimating distance measures calculated among individuals (default='Population').
- `mode` The particular genetic distance metric that you are going to use.

**Value**

A matrix with the genetic distances
Note

This function currently includes the following individual distance measures:

AMOVA  Inter-individual
Bray    Proportion of shared alleles

This function also supports genetic distances based upon stratum distances. The currently supported genetic distances are:

Euclidean  Euclidean frequency distance
cGD     Conditional Genetic Distance
Nei     Nei’s corrected genetic distance (1978)
Dps     Shared allele distance = 1 - Ps
Jaccard Jaccard set dissimilarity

Author(s)

Rodney J. Dyer <rjdyer@vcu.edu>

Examples

```r
AA <- locus( c("A","A") )
AB <- locus( c("A","B") )
BB <- locus( c("B","B") )
AC <- locus( c("A","C") )
AD <- locus( c("A","D") )
BC <- locus( c("B","C") )
BD <- locus( c("B","D") )
CC <- locus( c("C","C") )
CD <- locus( c("C","D") )
DD <- locus( c("D","D") )
loci <- c(AA, AB, AC, AD, BB, BC, BD, CC, CD, DD)
pops <- c(rep("A",5), rep("B",5))
df <- data.frame( Population=pops, TPI=loci)
genetic_distance(df, mode="AMOVA")
genetic_distance(df, mode="Dps")
```

description

This function is the main one used for estimating genetic diversity among strata. Given the large number of genetic diversity metrics, not all potential types are included.

Usage

```
genetic_diversity(x, stratum = NULL, mode = c("A", "Ae", "A95", "He", "Ho", "Fis", "Pe")[2])
```
Arguments

- **x** A data.frame object with `locus` columns.
- **stratum** The strata by which the genetic distances are estimated. This can be an optional parameter when estimating distance measures calculated among individuals (default='Population').
- **mode** The particular genetic diversity metric that you are going to use. The `gstudio` package currently includes the following individual distance measures:
  - **A** Number of alleles
  - **Ae** Effective number of alleles (default)
  - **A95** Number of alleles with frequency at least five percent
  - **He** Expected heterozygosity
  - **Ho** Observed heterozygosity
  - **Fis** Wright’s Inbreeding coefficient (size corrected).
  - **Pe** Locus polymorphic index.

Value

A data.frame with columns for strata, diversity (mode), and potentially P(mode=0).

Author(s)

Rodney J. Dyer <rjdyer@vcu.edu>

Examples

```r
AA <- locus( c("A","A" )
AB <- locus( c("A","B" )
BB <- locus( c("B","B" )
locus <- c(AA,AA,AA,BB,BB,BB,AB,AB,AA)
locus2 <- c(AB,BB,AA,BB,BB,AB,AB,AA,AA,BB)
df <- data.frame( Population, TPI=locus, PGM=locus2 )
genetic_diversity( df, mode="Ae")
```

---

**genetic_relatedness**  Estimates Pair-wise Relatedness

Description

This function returns single relatedness estimates as a pairwise matrix.

Usage

```r
genetic_relatedness(x, loci = NA, mode = c("Nason", "LynchRitland")[[1]],
 freqs = NA)
```
**Arguments**

- **x**
  A data.frame that has locus columns.

- **loci**
  The loci to use (if missing all loci are used).

- **mode**
  The kind of relatedness to be estimated. Currently Fij (the default) and Lynch-Ritland are available.

- **freqs**
  An optional data.frame (as returned by the function frequencies() with allele frequencies). If this is not provided, it will be estimated from all the data. This allows you to estimate relatedness among subsets of individuals using more global measures of relatedness.

**Value**

A matrix of pairwise relatedness estimates.

**Note**

This only works on diploid data and will return NA for any comparison of missing genotypes.

**Examples**

```r
loci <- c( locus(1:2), locus(c(2,2)), locus(1:2) )
genetic_relatedness( loci )
genetic_relatedness( loci, freqs = data.frame( Allele=c("1","2"), Frequency=c(0.5,0.5))
```

---

**genetic_structure**

Estimation of genetic structure statistics.

**Description**

This function estimates common genetic statistics.

**Usage**

```r
genetic_structure(x, stratum = "Population", mode = c("Gst", "Gst_prime", "Dest")[1], nperm = 0, size.correct = TRUE, pairwise = FALSE, locus)
```

**Arguments**

- **x**
  An object of type data.frame with at least a single column of type locus

- **stratum**
  The stratum to use as groupings (default=’Population’).

- **mode**
  Which statistic to estimate. Current options include:
  - **Gst** Nei’s Gst (not Berg and Hamrick)
  - **Gst_prime** Hedrick’s correction of Nei’s Gst for diverse loci
  - **Dest** Joost’s estimate

- **nperm**
  The number of permutations used to test the hypothesis that the parameter = 0.
Genotype Frequencies

Description

Provides a summary of genotype frequencies (observed and expected) from a vector of locus objects.

Usage

genotype_frequencies(x)

Arguments

x An object of type locus

Value

A data.frame with genotype, observed, and expected as counts.
geom_frequencies

Author(s)

Rodney J. Dyer <rjdyer@vcu.edu>

Examples

freqs <- c(0.55, 0.30, 0.15)
alleles <- c(LETTERS[1:3])
f <- data.frame(Locus="PGM", Allele=alleles, Frequency=freqs)
data <- make_population(f,N=20)
table(data$PGM)
genotype_frequencies( data$PGM )

geom_frequencies

Plot allele frequencies

Description

This function allows you to plot allele frequencies for several potential levels of stratum and loci.

Usage

geom_frequencies(x, ...)

Arguments

x The data frame with allele frequencies from frequencies.
...

Value

An object of type ggplot

Note

This function does a pretty good job of plotting but if you have too many loci and/or strata, you may have a difficult time getting it to look right. By default, the function does NOTHING to prevent you from making crappy plots with too many levels. To plot a reduced number of loci/strata, use a subset of your data.

Author(s)

Rodney J. Dyer <rjdyer@vcu.edu>
Examples

data(arapati)
freqs <- frequencies( arapati )
require(ggplot2)
ggplot() + geom_frequencies( freqs )
freqs <- frequencies( arapati, loci="EN", stratum="Cluster")
ggplot() + geom_frequencies( freqs ) + facet_grid(~Stratum)

geom_locus

Translate a vector of locus objects into a geom_bars layer

Description

This function takes a data frame containing genetic data and returns a geom_bars layer for ggplot integration.

Usage

geom_locus(mapping, data, ...)

Arguments

mapping The aesthetic mapping (e.g., which locus to use). Use aes(x=LOCUS_NAME) to specify which locus is being used.
data A data.frame containing one or more loci to be plot
... Added to geom_bar

Value

A formatted set of ggplot objects to be plot

Note

If using more than one stratum, use fill=STRATA_NAME for partitioning

Author(s)

Rodney J. Dyer <rjdyer@vcu.edu>

Examples

require(ggplot2)
loci <- c( locus(1:2), locus(2:3), locus(c(1,1)), locus(1:2) )
data <- data.frame( Population=c("A","A","B","B"), Locus=loci)
ggplot() + geom_locus( aes(x=Locus, fill=Population), data=data )
**geom_strata**

*Returns ggplot layer for population data*

**Description**

This function takes a data.frame of population data and returns a `geom_point()` layer for plotting. You can also indicate shape and color attributes in the mapping (through `aes`) that will be carried through.

**Usage**

```r
gem_strata(mapping = NULL, data = NULL, ...)```

**Arguments**

- `mapping` The aesthetic mapping, this MUST have values for x, y, and stratum
- `data` The data.frame from which the coordinates and other materials are to be pulled.
- `...` Other parameters submitted to `geom_point()`.

**Author(s)**

Rodney J. Dyer <rjdyer@vcu.edu>

**Examples**

```r
require(ggplot2)
data(arapat)

ggplot() + geom_strata(aes(x=Longitude, y=Latitude, stratum=Population), data=arapat) + coord_equal()
mapping <- aes(x=Longitude, y=Latitude, stratum=Population, color=Cluster, shape=Species)
s <- geom_strata(mapping, data=arapat)
ggplot() + s + coord_equal()
```

---

**geom_surface**

*Translate raster into ggplot object*

**Description**

This function takes a raster and translates it into a ggplot geometry layer for plotting.

**Usage**

```r
gem_surface(raster, ...)```
Arguments

raster An object of type raster
... Passed on to geom_tile as optional arguments.

Value

An object of type geom_tile

Author(s)

Rodney J. Dyer <rjdyer@vcu.edu>

grid.csv  

Description

Example data set for CDPop input

Gst  

Estimation Nei’s Gst parameter

Description

This function estimates Nei’s Gst parameter and potentially returns the components of it as well as the probability. The results are returned as a data.frame.

Usage

Gst(x, stratum = "Population", nperm = 0, size.correct = TRUE)

Arguments

x A vector of locus objects or a data.frame with locus objects.
stratum Either a vector of strata variables if x is a locus vector or the name of the column representing strata in x if it is a data.frame.
nperm The number of permutations to run for significance of the estimator.
size.correct A flag indicating that the estimate should be corrected for based upon sample sizes (default=TRUE).

Value

An data.frame with Gst, Ht, and Hs and optionally P
Author(s)
Rodney J. Dyer <rjdyer@vcu.edu>

Examples

```r
AA <- locus( c("A","A") )
AB <- locus( c("A","B") )
BB <- locus( c("B","B") )
locus <- c(AA,AA,AA,AA,BB,BB,BB,BB,AB,AB,AA)
Gst( locus, Population, nperm=99 )
locus2 <- c(AB,BB,AA,BB,AB,AB,AA,AA,BB)
df <- data.frame( Population, TPI=locus, PGM=locus2 )
Gst( df, nperm=99)
```

---

**Gst_prime**

*Estimation Nei’s G*<sub>s</sub>* G*<sub>t</sub>* parameter with Hedrick’s correction for allelic diversity*

---

**Description**

This function estimates the parameter (and potentially the confidence surrounding its value) for Gst. It is corrected by the diversity of the parameter as outlined by Hedrick.

**Usage**

```r
Gst_prime(x, stratum = "Population", nperm = 0, size.correct = TRUE)
```

**Arguments**

- `x` A vector of `locus` objects or a `data.frame` with `locus` objects.
- `stratum` Either a vector of strata variables if `x` is a `locus` vector or the name of the column representing strata in `x` if it is a `data.frame`.
- `nperm` The number of permutations to run for significance of the estimator.
- `size.correct` A flag indicating that the estimate should be corrected for based upon sample sizes (default=TRUE).

**Value**

An `data.frame` with Gst, H<sub>t</sub>, and H<sub>s</sub> and optionally P. If more than one locus is provided, then a 'multilocus' estimate is shown using the harmonic mean of individual locus Gst_prime values.

**Author(s)**
Rodney J. Dyer <rjdyer@vcu.edu>
Examples

```r
a1 <- sample( LETTERS[1:5], size=20, replace=TRUE)  
a2 <- sample( LETTERS[4:8], size=20, replace=TRUE)  
raw_alleles <- matrix( c(a1,a2), ncol=2, byrow=TRUE)  
locus <- locus( raw_alleles, type="column")  
Gst_prime( locus, Population )
```

```r
da1 <- sample( LETTERS[1:5], size=20, replace=TRUE)  
da2 <- sample( LETTERS[4:8], size=20, replace=TRUE)  
raw_alleles <- matrix( c(a1,a2), ncol=2, byrow=TRUE)  
locus2 <- locus( raw_alleles, type="column")  
df <- data.frame( Population, TPI=locus, PGM=locus2 )  
Gst_prime( df, nperm=99)
```

---

**He**

Estimate expected heterozygosity

**Description**

Returns the general expected heterozygosity parameter from the frequencies

**Usage**

```r
He(x, small.sample.correction = FALSE)
```

**Arguments**

- `x` Either a `data.frame` object with `locus` objects or a vector or `locus` objects.
- `small.sample.correction` Apply the $2N/(2N-1)$ correction to the data for small sample sizes.

**Value**

The expected heterozygosity as a numeric or a `data.frame` if several loci are passed.

**Author(s)**

Rodney J. Dyer <rjdyer@vcu.edu>

**Examples**

```r
loci <- c( locus( c("A","A") ), locus( c("A","A") ), locus( c("A","B") ) )
He( loci )
He( loci, small.sample.correction=TRUE )
```
**Ho**  
*Estimate observed heterozygosity*

**Description**

Returns the general observed heterozygosity parameter from the frequencies

**Usage**

```r
Ho(x)
```

**Arguments**

- `x`: An object of type `locus`

**Value**

The expected heterozygosity

**Author(s)**

Rodney J. Dyer <rjdyer@vcu.edu>

**Examples**

```r
loci <- c( locus( c("A","A") ), locus( c("A","A") ), locus( c("A","B") ) )
Ho( loci )
```

---

**indices**  
*Returns quickly indices for a matrix of size N*

**Description**

This is used in conjunction with apply to get around doing nested loops in R. This function is slightly faster than expand.grid. This function was lifted from a discussion on stackoverflow.

**Usage**

```r
indices(seq, only.unique = TRUE)
```

**Arguments**

- `seq`: The first Sequence.
- `only.unique`: Makes sure to return only i,j and not both [i,j] and [j,i] values

**Value**

A matrix of indicies
is.locus

An 'is-a' operator for locus

Description
A quick convenience function to determine if an object is inherited from the locus object.

Usage
is.locus(x)

Arguments
x An object to query

Value
A logical flag indicating if x is a type of locus

Author(s)
Rodney J. Dyer <rjdyer@vcu.edu>

Examples
loc1 <- locus( c("A","A") )
is.locus( loc1 )
is.locus( FALSE )
is.locus( 23 )

is.na.locus
Determines if locus is NA

Description
Convience function to determine if the locus object is NA

Usage
## S3 method for class 'locus'
is.na(x)

Arguments
x The locus object
Value

A logical flag indicating if \textit{x} is NA

Author(s)

Rodney J. Dyer <rjdyer@vcu.edu>

\begin{verbatim}
\textbf{is_frequency} \hspace{1cm} \textit{Is-A function for Allele Frequencies}
\end{verbatim}

Description

This is a function that returns a flag indicating that the object passed has the qualities of a \texttt{data.frame} created by the function \texttt{frequencies}.

Usage

\texttt{is_frequency(x)}

Arguments

\begin{itemize}
\item[] \texttt{x} \hspace{2cm} A \texttt{data.frame} potentially from \texttt{frequencies} function.
\end{itemize}

Value

A flag indicating it has the qualities of a \texttt{data.frame} representing allele frequencies

Author(s)

Rodney J. Dyer <rjdyer@vcu.edu>

\begin{verbatim}
\textbf{is_heterozygote} \hspace{1cm} \textit{Determine heterozygosity}
\end{verbatim}

Description

This function will determine the heterozygosity of one or more objects of type \texttt{locus}.

Usage

\texttt{is_heterozygote(x)}

Arguments

\begin{itemize}
\item[] \texttt{x} \hspace{2cm} A code object (single or vector)
**kronecker_delta**

**Value**
A logical flag indicating that there are at least two kinds of alleles present in the locus object.

**Note**
A haploid or NA locus is never heterozygous.

**Author(s)**
Rodney J. Dyer <rjdyer@vcu.edu>

**Examples**

```
loc <- locus(1:2)
is_heterozygote(loc)
```

---

**kronecker_delta**  
This function returns the kronecker operator for two loci.

**Description**
This is a quick utility function that provides the Kronecker operators for a pair of loci.

**Usage**

```
kronecker_delta(locus1, locus2)
```

**Arguments**

- `locus1`: An object of type 'locus'
- `locus2`: An object of type 'locus'

**Value**
a named vector of kronecker products.

**Note**
The Kronecker operator is a vector consisting of values related to the similarity of alleles among two loci. For the genotypes AiAj and AkAl the Kronecker deltas are \( \delta \) of the subscripted alleles are different and \( \delta \) if they are the same (e.g., \( \delta = 0 \) if heterzygote, \( \delta = 1 \) if homozygote)

**Author(s)**
Rodney J. Dyer <rjdyer@vcu.edu>
Examples

```r
loc1 <- locus( c(1,1) )
loc2 <- locus( c(1,2) )
loc3 <- locus( c(2,2) )
kronecker_delta( loc1, loc2 )
kronerKER_delta( loc1, loc3 )
kronerKER_delta( loc2, loc3 )
```

locus

*General constructor for locus object*

Description

This function creates an object of type locus, a fundamental type in the gstudio package. There are several kinds of loci that can be created.

Usage

```r
locus(x, type, phased = FALSE)
```

Arguments

- **x**
  - The data to be turned into a locus object
- **type**
  - An indication of what kind of data it is. By default this parameter is missing and this will cause the function to assume that every element of x is an allele in the genotype.
  - **blank** Default value, uses all passed items as alleles
  - **afp** Encoded as 0,1 for absence/presence of bands.
  - **column** Two columns of alleles
  - **separated** Pre-separated alleles (with ':').
  - **snp** Encoded by the number of minor alleles at the locus.
  - **zyme** Alleles like zymes (e.g., 12 for '1' and '2' alleles).
  - **snp_prob** A probabilistic snp call denoted as three posterior likelihoods for AA, AB, and BB as is often found in RAD-seq data.
- **phased** A flag indicating the the alleles should are of known gametic phase (default=FALSE).

Value

Either a single or vector of objects of type locus.

Author(s)

Rodney J. Dyer <rjdyer@vcu.edu>
Examples

```r
AA <- locus( c("A","A") )
AB <- locus( c("A","B") )
BB <- locus( c("B","B") )
AC <- locus( c("A","C") )
AD <- locus( c("A","D") )
BC <- locus( c("B","C") )
BD <- locus( c("B","D") )
CC <- locus( c("C","C") )
CD <- locus( c("C","D") )
DD <- locus( c("D","D") )
loci <- c(AA,AB,AC,AD,BC,BD,CC,CD,DD)
```

make_population

Makes a random population

Description

This function takes a frequencies() object and returns a data.frame with the appropriate stratum and loci derived randomly.

Usage

```r
make_population(x, N = 20)
```

Arguments

- `x` A data.frame as returned by the frequencies() function. For a single locus, it has columns "Allele" and "Frequency" for multiple loci, it has a "Locus" column, and if it is to be subdivided into strata, you have a "Strata" column.
- `N` The number of individuals to create (default = 20). This is a 'per-stratum' estimate so if you have K strata the result will be K*N individuals.

Value

A data.frame with individuals, loci, and potentially strata

Author(s)

Rodney J. Dyer <rjdyer@vcu.edu>

Examples

```r
define <- c(0.55, 0.30, 0.15, 0.34, 0.34, 0.32)
loci <- c(rep("TP1",3), rep("PGM",3))
alleles <- c(LETTERS[1:3],LETTERS[8:10])
f <- data.frame(Locus=loci, Allele=alleles, Frequency=define)
make_population(f,N=20)
```
Description

This function is mostly for simulation work and functions to produce a set of offspring for the individuals passed.

Usage

mate(mom, dad, N = 1)

Arguments

mom
This is the 'maternal' individual in that all the metadata in the data.frame that describes this individual will be transferred to the offspring.

dad
This is the 'paternal' individual and will contribute only half of its genetic complement to the offspring.

N
The number of offspring to produce.

Value

A data.frame of offspring.

Note

There are several 'hidden' things in this routine. First, if you do not pass a 'dad' object, it will assume you want a selfed offspring. Second, it will by default only make a single offspring. Next, if you have a column labelled "Sex" it will make a random selection of which sex each offspring should be and only mate the opposite sexes based upon the levels of the values in the Sex column. Finally, if there are columns ID and OffID in the mom, then all offspring will have the same ID as the mom but will have OffID equal to 1:N to conform with how the functions like paternity() operate. If you do not have ID and OffID then it will do nothing special. This can be more than one individual mom & dad, but if you pass several, they will all have the same number of offspring (if you only specify a single value of N) or different numbers of offspring (if N is passed as a vector and is of length equal to that of mom and dad)

Author(s)

Rodney J. Dyer <rjdyer@vcu.edu>

Examples

f <- data.frame(Allele=LETTERS[1:2], Frequency=c(0.5,0.5))
adults <- make_population(f,N=2)
adults
mate( adults[,1], adults[,2], N=10)
migrate

This function returns a data frame that has moved migrants

Description

This is a general function that moves migrants among strata according to either a constant migration rate or a set of migration rates defined by a migration matrix. All this does is adjust the stratum labels of individuals, no mating is conducted.

Usage

migrate(data, stratum = "Population", m = 0.1, relabel = TRUE)

Arguments

data A data.frame object with at least a stratum column. All other columns in the data frame are left untouched
stratum The column designating stratum.
m Either a rate of migration as a numeric OR a migration matrix whose columns and row names are the same as those in the passed stratum column.
relabel A flag (default TRUE) that relabels individuals after they have been moved between populations. Used for testing only.

Value

A data.frame with all the data the same except for the shuffled stratum.

Note

This is a simple transition matrix approach to migration. The migration rate times the population size MUST result in at least a whole number or else no migrants are produced. It uses round() to find the whole number of individuals to migrate. This means that if you have a migration rate that has less than one individual it will NEVER occur, not that it will occur at a less than one individual per generation frequency. You must adjust your migration rates directly for that (e.g., the user has the entire power, not the this function).

Author(s)

Rodney J. Dyer <rjdyer@vcu.edu>
minus_mom

Subtracts maternal component to offspring genotypes

Description

This function removes the female component to the offspring genotypes. It is one step in the 2gener analysis. The coding of the MomCol and the OffCol headers are specific so that this algorithm can differentiate between maternal individuals and offspring.

Usage

minus_mom(x, MomCol = "ID", OffCol = "OffID")

Arguments

x A data.frame with mother and offspring locus columns.
MomCol The name of the column indicating maternal ID. All offspring from a mother MUST have the same MomCol value.
OffCol The name of the column indicating the offspring ID number. It is required that maternal individuals have OffCol="0" indicating that this is the mother of those offspring.

Value

A data.frame object of only the offspring after removing the contribution of each maternal individual.

Author(s)

Rodney J. Dyer <rjdyer@vcu.edu>

Examples

AA <- locus( c("A","A") )
AB <- locus( c("A","B") )
BB <- locus( c("B","B") )
AC <- locus( c("A","C") )
AD <- locus( c("A","D") )
BC <- locus( c("B","C") )
BD <- locus( c("B","D") )
CC <- locus( c("C","C") )
CD <- locus( c("C","D") )
DD <- locus( c("D","D") )
loci <- c(AA,AB,AC,AD,BB,BC,BD,CD,CC,DD)
offID <- c(1,2,3,0,1,2,0,2)
df <- data.frame( ID=factor(momID), OffID=factor(offID), TPI=loci )
minus_mom( df )
**multilocus_diversity**

*Returns multilocus diversity*

**Description**

This function returns the unique population size of the population if all individuals were treated as multilocus individuals.

**Usage**

```r
call_multilocus_diversity(x)
```

**Arguments**

- `x`: A data frame with locus objects in it.

**Value**

The fraction of that data set that have unique multilocus genotypes.

**Author(s)**

Rodney J. Dyer <rjdyer@vcu.edu>

---

**optimal_sampling**

*Estimate optimal sampling allocation*

**Description**

This function returns a graphical representation of optimal sampling allocations given an estimate of the total number of samples you are looking at genotyping and the amount of expected differentiation among strata.

**Usage**

```r
call_optimal_sampling(N, phi)
```

**Arguments**

- `N`: The total sample size (number of strata * number of individuals per stratum)
- `phi`: An estimate of the level of differentiation.

**Value**

A ggplot object with a plot of the variance in phi due to different sample sizes with a region of +/- 10 phi value.
parent_finder

Author(s)
Rodney J. Dyer <rjdyer@vcu.edu>

Examples

```r
N <- 400
Phi <- 0.25
optimal_sampling(N, Phi)
```

parent_finder   This function finds a set of potential parents from an offspring.

Description
This is a general exclusion finder for a single parent. For dual parent exclusion, you can use the paternity() function.

Usage

```
parent_finder(df, ID = "ID", OffID = "OffID")
```

Arguments

df A data.frame of individuals adults individuals. This will be the population from which the identification of potential parentage will be examined.

ID The adult "ID" column designator (default="ID"). All adults have a unique ID designation. This will be how each adult is identified.

OffID This is a non-zero number for each offspring. The combination of ID and OffID should be able to be used to identify each offspring uniquely. All adults have OffID=0, by definition.

Value
A data.frame with the following columns. ID = the offspring id, OffID = the offspring OffID (these two will identify the offspring uniquely), ParentID = the putative parent identified, T = the multilocus transition probability associated with the offspring and parent.

Author(s)
Rodney J. Dyer <rjdyer@vcu.edu>
**partition**

**Subdivide, into a list of substrata**

**Description**

This function allows you to take a single Population object into a list of Population objects, one for each stratum.

**Usage**

```r
partition(x, stratum = "Population")
```

**Arguments**

- `x` Any object that can be column indexed by the function `names`
- `stratum` The column name of the stratum to partition on (default='Population').

**Value**

A partitioned list of objects, indexed by name

**Author(s)**

Rodney J. Dyer <rjdyer@vcu.edu>

**Examples**

```r
AA <- locus(c("A","A"))
AB <- locus(c("A","B"))
BB <- locus(c("B","B"))
locus <- c(AA,AA,AA,AA,BB,AB,AB,AB,AA)
locus2 <- c(AB,BB,AA,BB,AB,AB,AA,AA,BB)
df <- data.frame(Population, TPI=locus, PGM=locus2)
partition(df, stratum="Population")
```

---

**paternity**

**Estimates fractional paternity probability**

**Description**

This function estimates fractional paternity for offspring given a set of potential fathers.

**Usage**

```r
paternity(offspring, mother, fathers, ID = "ID", OffID = "OffID", strict = FALSE)
```
Arguments

offspring  A particular offspring. Often as a row from a data.frame with columns as loci and other meta data.

mother  The assumed mother of the offspring as a row from a data.frame with columns as loci and other meta data.

fathers  A data.frame of potential fathers.

ID  The name of the column where the dad's adult ID column is found in the data.frame of potential fathers.

OffID  The name of the column where the offspring ID is located. All offspring from a maternal individual should have have the same maternal ID to indicate which mother they are from but must also have a unique offspring ID.

strict  A flag (default=FALSE) to use strict paternity (e.g., where only one dad can be the father) versus fractional paternity (e.g., where potentially more than one father is assignable but whose likelihood is based upon their transition probabilities).

Value

A data.frame with indications of paternity by row. Columns will include ID, OffID, DadID, and potentially Fij.

Author(s)

Rodney J. Dyer <rjdyer@vcu.edu>

Examples

freqs <- c(1/3,1/3,1/3)
loci <- rep( paste("Loc","1:6,sep="-"), each=3)
alleles <- LETTERS[1:3]
f <- data.frame(Locus=loci, Allele=alleles, Frequency=freqs)
f
adults <- make_population(f,N=20)
adults
offs <- mate( adults[1,], adults[2,], N=10)
offs$OffID <- offs$ID
offs$MomID <- adults$ID[1]
paternity( offs, adults[1,], adults )

Pe  Polymorphic index for loci

Description

This function returns the effective number of alleles in a sample.
Usage

Pe(x)

Arguments

x A set of locus objects either as a vector or within a data.frame.

Value

The polymorphic index for the locus, \( \text{sum}(\pi(1-\pi)) \) as a numeric value or as a data.frame if multiple loci are passed.

Examples

```r
locus <- c( locus(1:2), locus(c(1,1)), locus(c(2,2)), locus(2:3) )
Pe(locus)
locus2 <- c( locus(1:2), locus(c(1,1)), locus(c(2,2)), locus(2:3) )
df <- data.frame( locus, locus2 )
Pe(df)
```

Description

This function takes some data and does permutations on it and passes it along to the indicated function. There are options for permuting genotypes or permuting alleles. Make sure you know the difference.

Usage

```r
permute_ci(x, stratum = NULL, allele_perm = TRUE, nperm = 99,
        FUN = NULL, replace = FALSE, ...)
```

Arguments

x The locus data being used.

stratum An optional argument if passed, permutes x among strata and gives the results to the function.

allele_perm A flag (default=TRUE) to permute alleles in the creating the new data set.

nperm The number of perms to use (default 99).

FUN the function to pass the permuted data to.

replace A flag indicating if randomization should be with replacement (default FALSE)

... Passed on as arguments to FUN.
Value

A vector of permuted response values from the function.

Author(s)

Rodney J. Dyer <rjdyer@vcu.edu>

pies_on_map  Plot overload

Description

Plots an allele frequencies object as bar or maps.

Usage

pies_on_map(x, stratum = "Population", locus = NA,
longitude = "Longitude", latitude = "Latitude", line.color = "black",
label = FALSE, palette = 8, ...)

Arguments

x  An object of type data.frame created from frequencies
stratum  The stratum to use for calculating frequencies (default 'Population')
locus  The name of the locus to use (default=NA)
longitude  The name of the Longitude data column (default 'Longitude')
latitude  The name of the Latitude data column (default 'Latitude')
line.color  An parameter indicating the color of the border of bars and pie wedges.
lable  A flag indicating that the stratum names will be printed in the map plots.
palette  The number of the brewer palette to use (default=8)
...  Ignored

Value

Nothing

Author(s)

Rodney J. Dyer <rjdyer@vcu.edu>
ploidy

**ploidy**

*Returns the number of alleles in a locus object*

**Description**

This function will a count of the number of alleles within a locus object for either a single locus or for a vector of loci.

**Usage**

```r
ploidy(x)
```

**Arguments**

- `x` A locus object (single or vector)

**Value**

A count of the number of alleles in the locus

**Author(s)**

Rodney J. Dyer <rjdyer@vcu.edu>

**Examples**

```r
loc <- locus(1:2)
ploidy(loc)
loci <- c(locus(1:2), locus(c(1,1)))
ploidy(loci)
```

---

plot.data.frame

*Plotting the locations of a ‘population’*

**Description**

Plots a data frame if that data frame has names indicated by the passed arguments stratum, Longitude, and Latitude. Otherwise it passes the object on to plot.default()

**Usage**

```r
## S3 method for class 'data.frame'
plot(x, stratum = "Population",
     longitude = "Longitude", latitude = "Latitude", map.source = "google",
     map.type = "terrain", color = "black", zoom = NA, ...)
```
plot.locus

Arguments

- **x**
  - An object of type `data.frame`
- **stratum**
  - The stratum to plot (default='Population').
- **longitude**
  - Name of the column to be used as a decimal longitude
- **latitude**
  - Name of the column to be used as decimal latitude
- **map.source**
  - Where the map should be retrieved from. Common options include
    - `google` Get the map from Google Maps (this is the default)
    - `osm` Open Street Map derived
- **map.type**
  - What kind of map to use. Current types include:
    - `satellite` A satellite image of the area.
    - `terrain` A stylized topological map (this is the default).
    - `road` The default google road map.
    - `hybrid` Mix of road and terrain maps.
- **color**
  - The color of the markers to be plot.
- **zoom**
  - The default zoom level when using google maps (default NA). If omitted the code will attempt to define the map by the bounding box defined by the coords
- **...**
  - Ignored

Value

- A ggplot object that will be plotted by default.

Author(s)

- Rodney J. Dyer <rjdyer@vcu.edu>

plot.locus  Overload plot function

Description

- This function is just for quick plotting of frequencies

Usage

```r
## S3 method for class 'locus'
plot(x, mode = c("bar", "pie")[1], ...)
```

Arguments

- **x**
  - A set of locus objects
- **mode**
  - The type of plot to create, bar or pie (bar is default).
- **...**
  - Ignored
Description

This function uses the ggmap package to grab a map of the area where the samples are located.

Usage

```r
population_map(coords, map.source = "google", map.type = "terrain", zoom = NA)
```

Arguments

- `coords`: A data frame that has Strata, Latitude, and Longitude
- `map.source`: Where the map should be retrieved from. Common options include:
  - `google`: Get the map from Google Maps (this is the default)
  - `osm`: Open Street Map derived
- `map.type`: What kind of map to use. Current types include:
  - `sattelite`: A satellite image of the area.
  - `terrain`: A stylized topological map (this is the default).
  - `road`: The default google road map.
  - `hybrid`: Mix of road and terrain maps.
- `zoom`: The default zoom level when using google maps (default NA). If omitted the code will attempt to define the map by the bounding box defined by the coords

Value

A map raster

Note

This is a bit of a rough attempt and should be used only sparingly. I have found that the ability to get Google Maps that cover all the sites is a bit difficult unless you play around with the zoom option. Google has fixed zoom levels so you may need to try this a few times before getting what you want.

Author(s)

Rodney J. Dyer <rjdyer@vcu.edu>
print.locus  
*Overload of print function for locus objects*

**Description**

Prints out the locus to stdout.

**Usage**

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

**Arguments**

- `x` : The locus object
- `...` : Other arguments passed to `print.default`.

**Value**

Nothing

**Author(s)**

Rodney J. Dyer <rjdyer@vcu.edu>

**Examples**

```r
loc <- locus( 1:2 )
print( loc )
```

---

rarefaction  
*Does allelic diversity rarefaction*

**Description**

This function subsamples the data with a smaller sample size and estimates the density of the diversity parameter.

**Usage**

```r
rarefaction(x, mode = "Ae", size = 0, nperm = 999)
```
Arguments

x The raw data as a locus vector.
mode The mode passed to allelic_diversity.
size The smaller sample size to use.
nperm The number of times to subsample the data (default 999)

Value

A vector of permuted values.

Author(s)

Rodney J. Dyer <rjdyer@vcu.edu>

read_population

Read a raw text file in and translate appropriate columns into genotypes

Description

The function reads in a text file and does the proper translations for genotypes and spatial coordinates.

Usage

read_population(path, type, locus.columns, phased = FALSE, sep = ",", header = TRUE, ...)

Arguments

path The path to the text file

An indication of what kind of loci that the data represent. The following kinds are recognized (n.b., if you have several types load them separately and merge them).

missing The default. This will cause read.population() to read each column as a single locus with one allele

aflp Encoded as 0,1 for absence/presence of bands.
column Two columns of alleles per locus.
separated Pre-separated alleles (with ":").
haploid One column per locus.
snp Encoded by the number of minor alleles at the locus.
zyme Alleles like zymes (e.g., 12 for '1' and '2' alleles).
genepop Import data that is in 'genepop' format.
cdpop Import genotypes encoded by CDPOP for subsequent analyses.
rel_lynch

Arguments

x 

A data.frame or vector that has locus objects in it. If you pass it a data.frame it will return the multilocus relatedness.

Value

A data.frame with locus columns pre-formatted.

Author(s)

Rodney J. Dyer <rjdyer@vcu.edu>

Description

This function estimates the allele-wise relatedness statistic from Lynch & Ritland (1999). It is very sensitive to the presence of rare alleles.

Usage

rel_lynch(x)

Details

This is an symmetric estimator using the formula:

\[
\hat{r}_{XY} = \frac{r_{XY} + r_{YX}}{2}
\]

where

\[
r_{XY} = \frac{1}{\sum_{i=1}^{L} w_i} \sum_{i=1}^{L} \frac{p_a(\delta_{bc} + \delta_{bd}) + p_b(\delta_{ac} + \delta_{ad}) - 4p_ap_b}{2p_ap_b}
\]

and
\[ w_i = \frac{(1 + \delta_{ab})(p_a + p_b) - 4 * p_a p_b}{2p_a p_b} \]

For missing genotypes, relatedness is not estimated for any pair-wise comparison. This does not constitute an error, specifically, but you should be careful of noting which comparisons are made from no or at least fewer than the full set of loci.

**Value**

A matrix of pairwise values

**Note**

For missing data and for data comparing heterozygotes at a 2-allele locus this estimator is undefined.

**Author(s)**

Rodney J. dyer <rjdyer@vcu.edu>

---

This estimates the Fij statistic from Nason allele-wise or for a whole locus

**Description**

This function estimates the allele-wise coancestry statistic fij from Nason. It can be used as a single locus or multilocus estimator if you provide the correct standardizations.

**Usage**

`rel_nason(x, allele = NA, as.relatedness = FALSE)`

**Arguments**

- `x` A vector of loci to use.
- `allele` The allele to estimate. If this is left blank, a locus-wide estimator is provided. That is the average of the allele-wise estimators standardized by the polymorphic index, Pe(\).
- `as.relatedness` Return r instead of Fij (default=FALSE)

**Value**

A matrix of pairwise values for either the allele or for the full locus.
Note
This estimator will use missing data but it treats it as if the frequency for each allele at an individual locus are equal to the population allele frequencies. This may influence your estimators.

Author(s)
Rodney J. Dyer <rjdyer@vcu.edu>

Description
This function estimates the allele-wise relatedness statistic from Queller & Goodnight (1989).

Usage
rel_queller(x)

Arguments
x A data.frame or vector that has locus objects in it. If you pass it a data.frame it will return the multilocus relatedness.

Details
This is an assymetric estimator using the formula:

\[ r_{XY} = \frac{\sum_{i=1}^{L}(\delta_{ac} + \delta_{ad} + \delta_{bd} - p_a - p_b - p_c - p_d)}{\sum_{i=1}^{L}(2 + \delta_{ab} + \delta_{cd} - p_a - p_b - p_c - p_d)} \]

Value
A matrix of pairwise relatedness values.

Note
For missing data and for data comparing heterozygotes at a 2-allele locus this estimator is undefined. Moreover, I do add zero to both the numerator.

Author(s)
Rodney J. dyer <rjdyer@vcu.edu>
**rel_ritland**

*This estimates the relatedness statistic from Ritland (1996)*

**Description**

This function takes a locus vector or a data.frame of locus objects and produces a pairwise distance matrix of relatedness values after Ritland (1996).

**Usage**

```
rel_ritland(x)
```

**Arguments**

- `x` A data.frame or vector that has locus objects in it. If you pass it a data.frame it will return the multilocus relatedness.

**Value**

A matrix of relatedness statistics.

**Author(s)**

Rodney J. dyer <rjdyer@vcu.edu>

---

**rep.locus**

*Replicate a locus*

**Description**

A quick replacement for rep so it does not replicate a locus object as a character so we can use functions like outer efficiently.

**Usage**

```
## S3 method for class 'locus'
rep(x, times, ...)
```

**Arguments**

- `x` An object of type character
- `times` The number of times to replicate this
- `...` Ignored
spiderplot_data

Value
A vector of locus objects

Author(s)
Rodney J. Dyer <rjdyer@vcu.edu>

Examples
loc1 <- locus(c("A","B"))
rep(loc1, times=4)

snp_prob.csv    SNP probability data

Description
Example data for probabilistically calling snps

spiderplot_data    Translate paternity output into segments for geom_segments

Description
This is a convenience function that takes the output of a paternity() function and the raw data.frame with the coordinates and turns it into a data.frame that can be used in geom_segments.

Usage
spiderplot_data(pat, df, ID = "ID", OffID = "OffID",
                 longitude = "Longitude", latitude = "Latitude")

Arguments

pat  The output from the paternity() function.
df   A data.frame with columns for the ID of the fathers and spatial coordinates.
ID   The name of the column in df that has the ID of the fathers (default="ID")
OffID The column name in df that has the offspring id value
longitude The label for the x-coordinate for the adults as it is found in the df object (default="Longitude")
latitude The label for the y-coordinate for the adults as it is found in the df object (default="Latitude")
strata_coordinates

Value

A data.frame with the segments correctly formatted for geom_segment.

Author(s)

Rodney J. Dyer <rjdyer@vcu.edu>

__strata_coordinates__

**Grab coordinates for strata**

Description

This function takes a , and a stratum and makes a data frame consisting of Stratum, Latitude, and Longitude for each stratum

Usage

```r
strata_coordinates(x, stratum = "Population", longitude = "Longitude",
latitude = "Latitude", as.SpatialPoints = FALSE, sort.output = FALSE)
```

Arguments

- `x` A data.frame object.
- `stratum` The name of the stratum to partition on (default="Population").
- `longitude` The column name of the longitude
- `latitude` The column name of the latitude
- `as.SpatialPoints` A flag indicating what kind of coordinates to return should be turned into a SpatialPoints object (TRUE) or as a data.frame (FALSE, the default)
- `sort.output` A flag indicating if the results should be sorted alphabetically (default=FALSE)

Value

A data frame, with Stratum Latitude and Longitude, summarized by center of each stratum.

Author(s)

Rodney J. Dyer <rjdyer@vcu.edu>
strata_distance

Return distances between strats

Description

This function takes a data.frame from the strata_coordinates function and returns the between strata distance matrix.

Usage

strata_distance(x, mode = c("Euclidean", "Circle") [2])

Arguments

x A data.frame object from strata_coordinates.
mode The type of distance to calculate. Currently there are two available types of physical distance, Euclidean (straight-line) and Great Circle (from the curvature of the earth)

Value

A data frame, with Stratum Latitude and Longitude, summarized by center of each stratum.

Note

This uses the mean radius of the earth to be 6371km as the latest estimate by USGS.

Author(s)

Rodney J. Dyer <rjdyer@vcu.edu>

summary.locus

Provides a summary of the locus object

Description

Provides an overload of the summary object so that when you put this into a data.frame and ask for a summary, it will provide you a summary of genotype counts.

Usage

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

---

summary.locus
to_fixed_locus

Arguments

object The locus object to summarize (typically a vector).

... Ignored

Value

A summary of the vector of locus objects in object

Author(s)

Rodney J. Dyer <rjdyer@vcu.edu>

---

to_fixed_locus returns string representation of locus for output like genepop/heirfstat

Description

This function

Usage

  to_fixed_locus(x, digits = NULL)

Arguments

x An object of type locus.

digits The number of digits that the alleles need to have

Value

A character representation of the locus with

Author(s)

Rodney J. Dyer <rjdyer@vcu.edu>
to_genepop  

Translates a data.frame

Description

This is a basic output file formatter for saving files necessary for saving output in textual formats.

Usage

to_genepop(df, stratum = "Population")

Arguments

df  The data.frame to be written to the output.
stratum  The stratum to use as "POP" (default="Population")

Value

A string representation of the data.frame formatted as

Author(s)

Rodney J. Dyer <rjdyer@vcu.edu>

Examples

A <- locus( c("1","1"))
B <- locus( c("1","2"))
C <- locus( c("2","2"))
loc1 <- c( A, A, B, B, B, B, C, C)
loc2 <- c( A, B, B, C, A, B, A, C)
pop <- data.frame( Population=c( rep("A",4),rep("B",4) ), loc1, loc2)
gp <- to_genepop( pop )
cat(gp)

________

to_mv  

Returns the passed items as multivariate data

Description

This function will provide the alleles within a locus object for either a single locus or for a vector of loci.
to_mv_freq

Usage

to_mv(x, ploidy, alleles, drop.allele = FALSE, ...)

## Default S3 method:
to_mv(x, ...)

## S3 method for class 'locus'
to_mv(x, ploidy = 2, alleles = NA, drop.allele = FALSE, ...)

## S3 method for class 'data.frame'
to_mv(x, ploidy = 2, alleles = NA,
       drop.allele = FALSE, ...)

Arguments

x A object to convert.
ploidy The number of alleles at an individuals locus (default=2)
alleles A set of alleles to make the matrix columns on.
drop.allele A flag indicating an allele should be dropped.
... Ignored

Value

The matrix representation of x.
Numerical value of x
Matrix representing the locus
A matrix

Author(s)

Rodney J. Dyer <rjdyer@vcu.edu>
Rodney J. Dyer <rjdyer@vcu.edu>

to_mv_freq Returns mv frequencies for stratum in long format

Description

This function takes a data.frame of data and returns a matrix of allele frequencies where each row is a stratum and each column is an allele frequency (all lumped together).

Usage

to_mv_freq(x, stratum = "Population")
to_structure

Translations a data.frame with loci into a textual STRUCTURE file.

Description

This is a basic converter that takes a data.frame with locus objects in it and returns a textual representation as a STRUCTURE input file.

Usage

to_structure(df, stratum = "Population")

Arguments

df The data.frame to be written to the output.
stratum The stratum to use as "POP" (default="Population")

Value

A string representation of the data.frame formatted for STRUCTURE

Author(s)

Rodney J. Dyer <rjdyer@vcu.edu>
transition_probability

Returns transition probability for offspring given one or more parents.

Description

This is a quick function used in paternity testing that provides the probability of transition for a locus of any ploidy level.

Usage

transition_probability(off, mom, dad, multilocus = TRUE)

Arguments

off A locus from the offspring
mom A locus from one parent
dad A locus from another parent
multilocus A flag indicating that a single multilocus estimate of the transition probability should be passed (the default action). If FALSE, the results will be returned as a vector of individual locus values.

Value

The numeric probability of the offspring given the parents

Author(s)

Rodney J. Dyer <rjdyer@vcu.edu>
write_population

Examples

```r
off <- locus(1:2)
mom <- locus(c(1,1))
dad1 <- locus(c(2,2))
dad2 <- locus(c(1,2))
dad3 <- locus(c(1,1))
transition_probability(off, mom, dad1)
transition_probability(off, mom, dad2)
transition_probability(off, mom, dad3)
```

write_population

**Writes data frame with genotypes and other data to a file**

Description

This is a basic output file formatter for saving files necessary for saving output in textual formats.

Usage

```r
write_population(df, file, mode = c("text", "genepop", "structure")][1],
                   stratum = NULL)
```

Arguments

- `df` The data.frame to be written to the output.
- `file` The path to the file you want to write to.
- `mode` Which format to use for writing data to file. At present there are
  - **text** Outputs all data columns with loci treated as colon separated. This is the default.
  - **genepop** Saves genetic data into GENEPOP format. You must specify a stratum.
  - **structure** Saves genetic data into the two-lined per individual STRUCTURE format. You must specify a stratum.
- `stratum` An optional argument if using genepop or structure formats. By default, the 'text' option writes all data to file.

Value

nothing.

Author(s)

Rodney J. Dyer <rjdyer@vcu.edu>
Description

An overload of the [ function for locus objects.

Usage

```r
## S3 method for class 'locus'
x[i]
```

Arguments

- `x` An object of type `locus`
- `i` The index of the allele to grab.

Value

The allele at the `i`th position.

Author(s)

Rodney J. Dyer <rjdyer@vcu.edu>

Examples

```r
loci <- c( locus(1:2), locus(1:2), locus( c(1,1) ) )
loci[2]
```
Index

*Topic data
  arapat, 9
  cornus_florida, 13
  data_2_column, 13
  data_aflp, 14
  data_separated, 14
  data_snp, 14
  data_zymelike, 14
  grid.csv, 33
  snp_prob.csv, 60
*Topic package
  gstudioMpackage, 3
  +.locus, 5
  -.locus, 6
  [.locus, 69
  A, 7
  Ae, 7
  alleles, 8
  arapat, 9
  as.data.frame.locus, 9
  as.list.locus, 10
  as.locus, 10
  bad_parents, 11
  c.locus, 12
  column_class, 12
  cornus_florida, 13
  data_2_column, 13
  data_aflp, 14
  data_separated, 14
  data_snp, 14
  data_zymelike, 14
  Dest, 15
  dist_amova, 16
  dist_bray, 17
  dist_cavalli, 18
  dist_cgd, 19
  dist_euclidean, 19
  dist_jaccard, 20
  dist_nei, 21
  dist_ss, 22
  exclusion_probability, 22
  Fis, 23
  frequencies, 24
  genetic_distance, 25
  genetic_diversity, 26
  genetic_relatedness, 27
  genetic_structure, 28
  genotype_frequencies, 29
  geom_frequencies, 30
  geom_locus, 31
  geom_strata, 32
  geom_surface, 32
  grid.csv, 33
  Gst, 33
  Gst_prime, 34
  gstudioMpackage, 3
  He, 35
  Ho, 36
  indices, 36
  is.locus, 37
  is_na.locus, 37
  is_frequency, 38
  is_heterozygote, 38
  kronecker_delta, 39
  locus, 11, 15, 25, 27, 28, 33, 34, 40
  make_population, 41
  mate, 42
  migrate, 43
  minus_mom, 44
multilocus_diversity, 45
optimal_sampling, 45
parent_finder, 46
partition, 47
paternity, 47
Pe, 48
permute_ci, 49
pies_on_map, 50
ploidy, 51
plot.data.frame, 51
plot.locus, 52
population_map, 53
print.default, 54
print.locus, 54
rarefaction, 54
read_population, 55
rel_lynch, 56
rel_nason, 57
rel_queller, 58
rel_ritland, 59
rep.locus, 59

snp_prob.csv, 60
spiderplot_data, 60
strata_coordinates, 61
strata_distance, 62
summary.locus, 62
to_fixed_locus, 63
to_genepop, 64
to_mv, 64
to_mv_freq, 65
to_structure, 66
transition_probability, 67
write_population, 68