Package: repvar (via r-universe)

August 31, 2024

Type Package

Title Extract Samples to Represent All Variables

Version 0.1.0

Depends R (>= 3.1.0)

BugReports https://github.com/zkamvar/repvar/issues

Imports stats, graphics

Maintainer Zhian N. Kamvar <zkamvar@gmail.com>

Description In population genetics, it's not uncommon to re-genotype sets of samples to use as positive controls in future studies or for diagnostic panels. To save cost, it's often desireable to have the minimum number of samples that represent all of the alleles in the data. This package provides a procedure that will select these samples with alternative options. The name 'repvar' stands for 'REPresent VARiables'.

License MIT + file LICENSE

Encoding UTF-8

LazyData true

Roxygen list(markdown = TRUE)

RoxygenNote 6.0.1

Suggests testthat, covr, tibble, tidyr, dplyr, purrr, knitr, rmarkdown

VignetteBuilder knitr

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

RemoteUrl https://github.com/zkamvar/repvar

RemoteRef HEAD

RemoteSha 07e2051f169c5df7f20f0fa2f57baa55455e8bdc

Contents

`monilinia` *Peach brown rot pathogen* Monilinia fructicola

Description

This is clone-censored microsatellite data for a population of the haploid plant pathogen *Monilinia fructicola* that causes disease within peach tree canopies (Everhart & Scherm, 2014). Entire populations within trees were sampled across 3 years (2009, 2010, and 2011) in a total of four trees, where one tree was sampled in all three years, for a total of 6 within-tree populations. Within each year, samples in the spring were taken from affected blossoms (termed "BB" for blossom blight) and in late summer from affected fruits (termed "FR" for fruit rot). From a total of 694 isolates, this data set represents 264 unique genotypes characterized uzing a set of 13 microsatellite markers comprised of 95 alleles.

Usage

```
data(monilinia)
```

Format

an integer matrix where rows represent individual samples and columns represent alleles at different loci where the locus name and fragment size are separated by a period.

References

SE Everhart, H Scherm, (2015) Fine-scale genetic structure of *Monilinia fructicola* during brown rot epidemics within individual peach tree canopies. *Phytopathology* **105**:542-549 doi: 10.1094/PHYTO-03-14-0088-R

Examples

```
data(monilinia)
(i <- rpv_indices(monilinia))
```

repvar *repvar.*

Description

This package allows you to find the minimum set of samples that represents all non-zero variables in a binary matrix.

Details

Minimum set of samples can be found with the function `rpv_find()`. This function will shuffle your data set and pass it to `rpv_indices()`. From this, you will have a list of sample names.

Because there may be several combinations of samples that represent all variables, the function `rpv_stats()` can be used to calculate entropy statistics over these variables.

If you want to visualize your data set, you can use `rpv_image()`

rpv_find *Iteratively find minimum set of samples by shuffling rows*

Description

Because `rpv_indices()` is deterministic, it may not present the minimum set that represents all variables. This procedure automates the process of randomly sampling the rows in the incoming matrix without replacement to find a minimum set.

Usage

```
rpv_find(tab, n = 10, sort = TRUE, cut = FALSE, progress = TRUE)
```

Arguments

`tab`	a numeric matrix with rownames
`n`	the number of permutations to perform
`sort`	when `TRUE` (default), the returned list will be sorted in order of number of samples.
`cut`	when `TRUE`, only the results with the minimum number of samples will be returned.
`progress`	when `TRUE`, a progress bar will be displayed.

Value

a list of character vectors

Examples

```
data(monilinia)
# Iterate over the data 100 times and return only the minimum values
set.seed(2018)
rpv_find(monilinia, n = 100, cut = TRUE, progress = FALSE)

# This is a random process and will not always return the same values
set.seed(201)
rpv_find(monilinia, n = 100, cut = TRUE, progress = FALSE)
```

---

rpv_image *Create an image of the data matrix*

Description

Create an image of the data matrix

Usage

```
rpv_image(tab, f = NULL, highlight = NULL, newplot = TRUE,
  col = c("#A6CEE3", "#1F78B4"), idcol = c("#FFFF99", "#B15928"))
```

Arguments

tab	a numeric matrix
f	a factor that is the same length as the number of columns in tab. this is used to split the matrix up by groups for analysis.
highlight	a character vector specifing which row names or indices to highlight.
newplot	When TRUE (default), The image will not over-write the previous image. Turn this off if you want to use multi-panel plotting.
col	a two-color vector for the values of the matrix.
idcol	a two-color vector for the highlight values.

Details

This function creates a plot that will allow you to visualize a matrix, optionally overlaying data. Note: the values here represent the presence/absence of a variable, but does not represent the dosage.

Value

NULL, invisibly

Examples

```
data(monilinia)
loci <- sapply(strsplit(colnames(monilinia), "[.]"), "[", 1)
rpv_image(monilinia, f = loci)
rpv_image(monilinia, f = loci, highlight = rpv_indices(monilinia))
```

rpv_indices *Get minimum set of individual indices to represent all alleles in a population*

Description

Get minimum set of individual indices to represent all alleles in a population

Usage

```
rpv_indices(tab)
```

Arguments

tab	an n x m matrix of individuals in rows and alleles in columns.

Value

a vector of integers representing row indices in the tab

Examples

```
data(monilinia)
i <- rpv_indices(monilinia)
i
all(colSums(monilinia[i, ], na.rm = TRUE) > 0)
```

rpv_stats *Calculate Entropy Statistics*

Description

Because it's possible to have multiple results with a minimum number of samples, one way of assessing their importance is to calculate how distributed the alleles are among the samples. This can be done with entropy statistics

Usage

```
rpv_stats(tab, f = NULL)
```

Arguments

tab	a numeric matrix
f	a factor that is the same length as the number of columns in tab. this is used to split the matrix up by groups for analysis.

Details

This function caluclates four statistics from your data using variable counts.

- eH: The exponentiation of shannon's entropy: `exp(sum(-x * log(x)))` (Shannon, 1948)
- G : Stoddart and Taylor's index, or inverse Simpson's index: `1/sum(x^2)` (Stoddart and Taylor, 1988; Simpson, 1949)
- E5: Evenness (5) the ratio between the above two estimates: `(G - 1)/(eH - 1)` (Pielou, 1975)
- lambda: Unbiased Simpson's index: `(n/(n-1))*(1 - sum(x^2))`
- missing: the percent missing data out of the total number of cells.

Both G and eH can be thought of as the number of equally abundant variables to acheive the same observed diversity. Both G and eH give different weight to variables based on their abundance, so we use evenness to describe how uniform this distribution is.

Note that this version of Evenness is different than Shannon's Evenness, which is `H/ln(S)` where S is the number of variables (in our case).

If a vector of factors is supplied, the columns of the matrix is first split by this factor and each statistic calculated on each level.

Value

a data frame with three columns: `eH`, `G`, `E5`, `lambda`, and `missing`

Note

The calculations within this function are derived from the vegan and poppr R packages.

References

Claude Elwood Shannon. A mathematical theory of communication. Bell Systems Technical Journal, 27:379-423,623-656, 1948

Simpson, E. H. Measurement of diversity. Nature 163: 688, 1949 doi:10.1038/163688a0

J.A. Stoddart and J.F. Taylor. Genotypic diversity: estimation and prediction in samples. Genetics, 118(4):705-11, 1988.

E.C. Pielou. Ecological Diversity. Wiley, 1975.

Examples

```
# Calculate statistics for the whole data set ----------------------------

data(monilinia)
rpv_stats(monilinia)

# Use a grouping factor for variables ------------------------------------
# Each variable in this data set represents and allele that is one of
# thirteen loci. If we wanted a table across all loci individually, we can
# group by locus name.

f <- gsub("[.][0-9]+", "", colnames(monilinia))
```

```
f <- factor(f, levels = unique(f))
colMeans(emon <- rpv_stats(monilinia, f = f)) # average entropy across loci
emon

# calculating entropy for minimum sets -------------------------------------

set.seed(1999)
i <- rpv_find(monilinia, n = 150, cut = TRUE, progress = FALSE)
colMeans(emon1 <- rpv_stats(monilinia[i[[1]], ], f = f))
colMeans(emon2 <- rpv_stats(monilinia[i[[2]], ], f = f))
```

Index