Package: coreCollection (via r-universe)

September 13, 2024

Type Package

Title Core Collection

Version 0.9.5

Description Create a custom sized Core Collection based on a distance matrix and applying the A-NE (accession nearest entry), E-NE (entry nearest entry) or E-E (entry entry) method as introduced in Jansen and van Hintum (2007) <doi:10.1007/s00122-006-0433-9> and further elaborated on in Odong, T.L. (2012) <https://edepot.wur.nl/212422>. Optionally a list of preselected accessions to be included into the core can be set. For each accession in the computed core, if available nearby accessions are retrievable that can be used as an alternative.

License GPL (>=2)

Encoding UTF-8

Depends R (>= 3.5)

LinkingTo Rcpp

Imports Rcpp (>= 1.0.0), R6 (>= 2.4.0), methods

Collate coreCollection.package.R coreCollection.R coreSelection.R RcppExports.R

Suggests testthat (>= 3.0.0), vcfR, adegenet, ggfortify

RoxygenNote 7.2.2

NeedsCompilation yes

Author Matthijs Brouwer [aut, cre] (<https://orcid.org/0000-0001-8183-0484>), Reinhoud Blok, de [ctb]

Maintainer Matthijs Brouwer <matthijs.brouwer@wur.nl>

Config/testthat/edition 3

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

RemoteUrl https://github.com/pbr/corecollection

RemoteRef HEAD

RemoteSha c45e8ccf326fe794aca344e05ec42d119ed032c4

Contents

`coreCollection-package`

The coreCollection package

Description

This package can be used to create a CoreCollection object.

Author(s)

Matthijs Brouwer <matthijs.brouwer@wur.nl>

References

Odong, T.L. (2012) Quantative methods for sampling of germplasm collections - Getting the best out of molecular markers when creating core collections. PhD diss., Wageningen University and Research, Wageningen, The Netherlands. `http://edepot.wur.nl/212422`

Jansen, J & Hintum, Theo. (2007) Genetic distance sampling: A novel sampling method for obtaining core collections using genetic distances with an application to cultivated lettuce. TAG. Theoretical and applied genetics. Theoretische und angewandte Genetik. 114. 421-8. 10.1007/s00122-006-0433-9

See Also

- **vcfR** provides a suite of tools for input and output of variant call format (VCF) files, manipulation of their content and visualization.
- **adegenet** provides the `genlight` class for genome-wide SNP data, and includes a method to create a distance matrix.

Other core collection: `CoreCollection()`

`CoreCollection` *The CoreCollection class*

Description

R6 class for creating a core collection based on the provided `distanceMatrix`, required size of the core `n` and optionally a set of `preselected` accessions to be included into the core.

Usage

```
CoreCollection(
  distanceMatrix,
  n,
  preselected = c(),
  coreSelectMethod = "A-NE",
  adjustedGroupMethod = "split",
  algorithm = "randomDescent",
  seed = NULL
)
```

Arguments

`distanceMatrix` A distance matrix; can be either a matrix or a dist

`n` The number of items in the core

`preselected` An optional list of preselected accessions to be included in the core collection; the provided accessions should occur in the labels or rownames of the provided distanceMatrix

`coreSelectMethod`
The method for computing core accessions within the groups: `A-NE` (accession nearest entry), `E-NE` (entry nearest entry) or `E-E` (entry entry)

`adjustedGroupMethod`
The method to handle adjusting groups when multiple preselected accessions occur within a single group: `split` to just split the initial groups with multiple accessions or `recompute` to recompute the division of accessions over the groups.

`algorithm` Algorithm applied to compute a solution: currently, only `randomDescent` is available

`seed` The seed used when generating the core collection. If no seed is provided, a random seed is chosen and each time the `recompute()` method is called on the object, a new seed will be used.

Details

Based on a provided `distanceMatrix` and required number `n` of accessions within the core, a random set of accessions is created, implicitly dividing the full population into initial groups based on the nearest randomly chosen random accession. If a set of `preselected` accessions is provided, this initial division is adjusted using the `adjustedGroupMethod`. Then, using the `coreSelectMethod` in the `algorithm`, the core accessions within these groups are calculated, resulting in the final core collection.

Fields

`adjustedBasedGroups` A list describing the initial random division of all accessions into groups, adjusted for the set of `preselected` accessions by using the defined `adjustedGroupMethod`.

`adjustedGroupMethod` The method to handle adjusting groups when multiple preselected accessions occur within a single group.

`adjustedSelected` A data.frame representing the intial random selection of accesions, adjusted for the set of `preselected` accessions by using the defined `adjustedGroupMethod`, with the accession names as labels and the following columns:

- `contains`: the (positive) number of accessions that have this accessions as the closest random selected accession
- `preselects`: the number of these closest accessions that were preselected
- `preselected`: a boolean indicating if the random selected accession was preselected
- `random`: a boolean indiciating if the selected accesion was initially randomly chosen or introduced later by the applied `adjustedGroupMethod`.

`algorithm` The applied algorithm to compute the solution.

`core` A data.frame representing the core collection with the accession names as labels and in the first and only column a boolean value indicating whether or not the accession was preselected.

`coreSelectMethod` The applied method to select the core accessions based on the computed `adjustedBasedGroups`.

`distanceMatrix` The distance matrix; this will allways be a dist object.

`n` The required core size

`pop` A data.frame representing the whole collection with the accession names as labels and in the first and only column:

- `result`: a string describing if the accession is marked as `other` or as included in the `core`, and if in the `core` because it was `preselected` or because of the applied `coreSelectMethod`.

`preselected` The list of preselected accessions.

`randomBasedGroups` A list with the initial division into groups based on the initial random selection of accessions described by `randomSelected`. Each item describes all accessions that have the random selected accesion from the label as the nearest neighbour, including the random selected accession.

`randomSelected` A data.frame representing the intial random selection of accesions with the accession names as labels and the following columns:

- `contains`: the (positive) number of accessions that have this accessions as the closest random selected accession
- `preselects`: the number of these closest accessions that were preselected
- `preselected`: a boolean indicating if the random selected accession was preselected
- `random`: a boolean indiciating if the random selected accesion was randomly chosen. This will always be TRUE for this field, but including this column makes the output comparable with `adjustedSelected`.

`seed` The last applied seed for the randomizer. This will only change when the `recompute()` method is called and no initial `seed` is defined.

Methods

`alternativeCore(n)` The `nth` alternative core with `n` a positive integer. Provides for each accession in the core, if available, the `nth` nearest accession from within the same group as an alternative.

`clone(deep = FALSE)` The default R6Class clone method.

`initialize(distanceMatrix, n, preselected, coreSelectMethod, adjustedGroupMethod, algorithm, seed)` Initialisation of the object, is called automatically on creation or recomputing.

`measure(coreSelectMethod)` The measure for the provided `coreSelectMethod`. If no value is provided, the current selected `coreSelectMethod` is used. The measure is used by the algorithm to compute the core collection.

`measures()` A data.frame with the available `coreSelectMethods` as labels and in the first and only column the measures for these methods.

`recompute()` Recompute the core collection: If on initialisation of the object a seed was provided, this same seed will be applied and therefore the same core collection will be created. Otherwise, a new seed is generated, resulting in a new core.

`print()` Create a summary of the core collection object, same as `summary()`.

`summary()` Create a summary of the core collection object, same as `print()`.

See Also

Other core collection: `coreCollection-package`

Index