Package 'triversity'

October 14, 2022

Title Diversity Measures on Tripartite Graphs

Version 1.0

Author Robin Lamarche-Perrin [aut, cre]

Maintainer Robin Lamarche-Perrin <Robin.Lamarche-Perrin@lip6.fr>

Description Computing diversity measures on tripartite graphs. This package first implements a parametrized family of such diversity measures which apply on probability distributions. Sometimes called ``True Diversity'', this family contains famous measures such as the richness, the Shannon entropy, the Herfindahl-Hirschman index, and the Berger-Parker index. Second, the package allows to apply these measures on probability distributions resulting from random walks between the levels of tripartite graphs. By defining an initial distribution at a given level of the graph and a path to follow between the three levels, the probability of the walker's position within the final level is then computed, thus providing a particular instance of diversity to measure.

Depends R (>= 3.2.3), Matrix, data.tree

License GPL-3 | file LICENSE

Encoding UTF-8

RoxygenNote 6.0.1

NeedsCompilation no

Repository CRAN

Date/Publication 2017-10-11 17:30:09 UTC

R topics documented:

get_conditional_diversity_from_transition
Compute the conditional diversity of a transition matrix.

Description

get_conditional_diversity_from_transition computes the geometric means of diversity values associated to the lines of the input transition matrix, while weighting these values according to an optional input distribution. This hence allows to compute conditional diversity values associated to the matrix.

Usage

```
get_conditional_diversity_from_transition(transition, distribution = NULL,
  order = NULL, measure = NULL)
```

Arguments

transition	A matrix of floats in [0,1], with all lines summing to 1, giving the transition matrix from which the conditional diversity values are computed.
distribution	A vector of floats in [0,1] and summing to 1 giving the probability distribution that is used to weight the diversity values when computing their geometric means. It should hence contain as many values as there are rows in the input transition. If not specified, this distribution is assumed uniform.
order	A vector of positive floats (possibly including Inf) giving the orders of the diversity measures to be computed. If neither order nor measure is specified, a predefined list of 8 diversity measures is computed.
measure	A vector of strings giving the names of the diversity measures to compute. Possible values are richness, entropy, herfindahl, and bergerparker.

Value

A vector of positive floats giving the conditional diversity values of the input transition matrix, that is the geometric means of the diversity values associated to its rows.

Examples

```
transition <- matrix (c (1/3, 1/3, 1/3, 0.9, 0.1, 0), nrow=2, ncol=3, byrow=TRUE)
get_conditional_diversity_from_transition (transition, order=c(0,Inf), measure='entropy')
get_conditional_diversity_from_transition (transition, distribution=c(0.75,0.25))
```

get_distribution_from_path

Compute the probability distribution associated to a random walk following a path between the levels of a tripartite graph.

Description

get_distribution_from_path computes the probability distribution of a random walk within the different levels of the input tripartite graph. It starts at a given level with an initial probability distribution, then randomly follows the links of the graph between the different levels according to the input path, then stops at the last specified level.

Usage

```
get_distribution_from_path(tripartite, path, initial_distribution = NULL,
  initial_node = NULL)
```

Arguments

tripartite	A tripartite graph obtained by calling the get_tripartite function.
path	A vector of integers in {1, 2, 3} giving the path of the random walk between the different levels of the input tripartite graph. This path can be as long as wanted. Two successive levels should always be adjacent, that is the random walk cannot go from level 1 to level 3 (or conversely) without first going through level 2.
initial_distribution	A vector of floats in [0,1] and summing to 1 giving the probability distribution to start with at the first level of the input path. It should hence contain as many values as there are nodes in the corresponding level. If not specified, this distribution is assumed uniform.
initial_node	A string giving the name of a node in the first level of the input path. This node is then considered to have probability one, thus being equivalent to specifying an initial_distribution with only zeros except for one node. If not specified, no such node is defined and the initial distribution is assumed uniform.

Value

A vector of floats in [0,1] and summing to 1 giving the probability distribution of the random walk when arriving at the last level, after having followed the input path within the different levels of the graph.

Examples

```
data (tripartite_example)
tripartite <- get_tripartite (data=tripartite_example)

get_distribution_from_path (tripartite, path=c(2,1,2,3))
```

```
get_distribution_from_path (tripartite, path=c(2,1,2,3), initial_distribution=c(0.25,0,0.25,0.5))
get_distribution_from_path (tripartite, path=c(2,1,2,3), initial_node='i2')
```

`get_diversity_from_distribution`
Compute the diversity of a probability distribution.

Description

`get_diversity_from_distribution` computes diversity values associated to an input probability `distribution`. The implemented diversity measures all belong to the parametrized family of "True Diversity" measures. They can either be specified by their diversity `order` in `[0,Inf[` or by their `measure` name when it corresponds to classical instances such as the richness, the Shannon entropy, the Herfindahl-Hirschman index, or the Berger-Parker index.

Usage

```
get_diversity_from_distribution(distribution, order = NULL, measure = NULL)
```

Arguments

`distribution`	A vector of floats in `[0,1]` and summing to `1` giving the probability distribution whose diversity is measured.
`order`	A vector of positive floats (possibly including `Inf`) giving the orders of the diversity measures to be computed. If neither `order` nor `measure` is specified, a predefined list of 8 diversity measures is computed.
`measure`	A vector of strings giving the names of the diversity measures to compute. Possible values are `richness`, `entropy`, `herfindahl`, and `bergerparker`.

Value

A vector of positive floats giving the diversity values of the input probability distribution.

Examples

```
distribution <- c (1/4, 1/2, 1/12, 2/12)

get_diversity_from_distribution (distribution)
get_diversity_from_distribution (distribution, order=c(0,Inf), measure='entropy')
```

`get_diversity_from_path` *Compute the diversity associated to a random walk following a path between the levels of a tripartite graph.*

Description

`get_diversity_from_path` computes diversity values of the probability distribution of a random walk following a `path` between the different levels of the input `tripartite` graph. It starts at a given level with an initial probability distribution, then randomly follows the links of the graph between the different levels according to the input `path`, then stops at the last specified level. The implemented diversity measures all belong to the parametrized family of "True Diversity" measures. They can either be specified by their diversity `order` in `[0,Inf[` or by their `measure` name when it corresponds to classical instances such as the richness, the Shannon entropy, the Herfindahl-Hirschman index, or the Berger-Parker index.

Usage

```
get_diversity_from_path(tripartite, path, conditional_path = NULL,
  initial_distribution = NULL, initial_node = NULL, order = NULL,
  measure = NULL)
```

Arguments

`tripartite`	A tripartite graph obtained by calling the `get_tripartite` function.
`path`	A vector of integers in `{1, 2, 3}` giving the path of the random walk between the different levels of the input `tripartite` graph. This path can be as long as wanted. Two successive levels should always be adjacent, that is the random walk cannot go from level `1` to level `3` (or conversely) without first going through level `2`.
`conditional_path`	A vector of integers in `{1, 2, 3}` eventually giving another path to compute conditional diversity values instead of regular diversity values. When specified, this conditional path is first used to initiate the random walk. The resulting probability distribution is then used to weight the individual diversity values obtained on the input `path` when computing their geometric means (see `get_conditional_diversity_from_transition`). This path can be as long as wanted. The last level of the conditional path should be the same as the first level of the input `path`. Moreover, two successive levels should always be adjacent, that is the random walk cannot go from level `1` to level `3` (or conversely) without first going through level `2`.
`initial_distribution`	A vector of floats in `[0,1]` and summing to `1` giving the probability distribution to start with at the first level of the input `path`, or at the first level of the input `conditional_path` if specified. It should hence contain as many values as there are nodes in the corresponding level. If not specified, this distribution is assumed uniform.

`initial_node` A string giving the name of a node in the first level of the input `path`, or at the first level of the input `conditional_path` if specified. This node is then considered to have probability one, thus being equivalent to specifying an `initial_distribution` with only zeros except for one node. If not specified, no such node is defined and the initial distribution is assumed uniform.

`order` A vector of positive floats (possibly including `Inf`) giving the orders of the diversity measures to be computed. If neither `order` nor `measure` is specified, a predefined list of 8 diversity measures is computed.

`measure` A vector of strings giving the names of the diversity measures to compute. Possible values are `richness`, `entropy`, `herfindahl`, and `bergerparker`.

Value

A vector of positive floats giving the diversity values (or conditional diversity values) of the random walk following the input `path`.

Examples

```
data (tripartite_example)
tripartite <- get_tripartite (data=tripartite_example)


# COMPUTING DIFFERENT DIVERSITY VALUES FOR A GIVEN PATH

# Herfindahl-Hirschman index of nodes in level 3 wrt nodes in level 1
get_diversity_from_path (tripartite, path=c(1,2,3), measure='herfindahl')
1 / get_diversity_from_path (tripartite, path=c(1,2,3), order=2)

# Shannon entropy of nodes in level 3 wrt nodes in level 1
get_diversity_from_path (tripartite, path=c(1,2,3), measure='entropy')
log2 (get_diversity_from_path (tripartite, path=c(1,2,3), order=1))

# Some other diversity values of nodes in level 3 wrt nodes in level 1
get_diversity_from_path (tripartite, path=c(1,2,3), order=c(1,2,Inf),
                         measure=c('richness','bergerparker'))

# Eight of the main diversity values of nodes in level 3 wrt nodes in level 1
get_diversity_from_path (tripartite, path=c(1,2,3))


# SPECIFYING THE INITIAL DISTRIBUTION

# Diversity of nodes in level 3 wrt nodes in level 1 (with non-uniform weights)
get_diversity_from_path (tripartite, path=c(1,2,3), initial_distribution=c(0.75,0.25))

# Individual diversity of nodes in level 3 wrt node 'u1' in level 1
get_diversity_from_path (tripartite, path=c(1,2,3), initial_node='u1')
get_diversity_from_path (tripartite, path=c(1,2,3), initial_distribution=c(1,0))


# COMPUTING THE MEAN OF INDIVIDUAL DIVERSITES
```

```
# Mean of individual diversities of nodes in level 3 wrt nodes in level 2 (with
# uniform weights)
get_diversity_from_path (tripartite, path=c(2,3), conditional_path=c(2))

# Mean of individual diversities of nodes in level 3 wrt nodes in level 2 (weighted
# according to the path from level 1 to level 2, with a uniform distribution in level 1)
get_diversity_from_path (tripartite, path=c(2,3), conditional_path=c(1,2))

# Mean of individual diversities of nodes in level 3 wrt nodes in level 2 (weighted
# according to the path from level 1 to level 2, with only node 'u1' in level 1)
get_diversity_from_path (tripartite, path=c(2,3), conditional_path=c(1,2),
                         initial_node='u1')
```

`get_transition_from_path` *Compute the transition matrix of a random walk following a path between the levels of a tripartite graph.*

Description

`get_transition_from_path` computes the transition matrix of a random walk following a `path` between the different levels of the input `tripartite` graph.

Usage

```
get_transition_from_path(tripartite, path)
```

Arguments

`tripartite`	A tripartite graph obtained by calling the `get_tripartite` function.
`path`	A vector of integers in {1, 2, 3} giving the path of the random walk between the different levels of the input `tripartite` graph. This path can be as long as wanted. Two successive levels should always be adjacent, that is the random walk cannot go from level 1 to level 3 (or conversely) without first going through level 2.

Details

Note that the tripartite graph structure implemented in this package stores in memory any computed transition matrix to avoid redundant computation in the future. Hence, the first execution of `get_transition_from_path`, or of any other function that builds on it, can be much slower than latter calls. The transition matrices are stored within a `data.tree` in the input `tripartite` variable (see `tripartite$transitions`).

Value

A matrix of floats in [0,1], with all lines summing to 1, giving the transition matrix of the random walk following the input path.

Examples

```
data (tripartite_example)
tripartite <- get_tripartite (data=tripartite_example)

get_transition_from_path (tripartite, c(2,1,2,3))
```

---

get_tripartite *Build a properly-structured tripartite graph from raw data.*

Description

get_tripartite builds a properly-structured tripartite graph from a file or from a data.frame containing raw data. This object can then be used by the other functions of this package. The structure of the input data and of the resulting structure is detailed below.

Usage

```
get_tripartite(filename = NULL, data = NULL)
```

Arguments

filename	A string giving the path to the file containing the raw data. The input file should contain at least four columns, separated by spaces. Each row gives a link between two nodes belonging to two different levels of the tripartite graph. The first column gives the level of the first node (any integer in {1, 2, 3}) and the second column gives its name (any character string). Similarly, the third and fourth columns give the level and the name of the second node. Note that no link is allowed between level 1 and level 3. The file can eventually contain a fifth column giving the weights of the links (any positive integer or float value).
data	A data.frame containing the raw data. This data.frame should have the same structure than the one described above for the case of an input file: four columns indicating the levels and the names of the two nodes constituting the link, and an optional fifth column for its weight. At least filename or data should be specified, but not both at the same time.

Value

A properly-structured tripartite graph that can be used by the other functions of the `triversity` package.

The resulting object encodes the names of the nodes at the three levels of the tripartite graph, as well as the transition matrices of random walks following different paths between levels (encoded as sparse matrices of floats in [0,1], with all rows summing to 1). These transition matrices are then used by functions such as `get_distribution_from_path` to compute the probability distributions of such random walks, or such as `get_diversity_from_path` to compute the diversity of these distributions.

Assuming the object returned by `get_tripartite` is stored in the `tripartite` variable, then:

- `tripartite$nodes` is a list of string vectors giving the names of the nodes constituting the three levels of the tripartite graph (resp. `tripartite$nodes$level1`, `tripartite$nodes$level2`, and `tripartite$nodes$level3`).
- `tripartite$transitions` is a `data.tree` whose nodes each contains the transition matrix of the corresponding random walk. For example, `tripartite$transitions$level1$level2$mat` gives the transition matrix from level 1 to level 2.

Examples

```
data (tripartite_example)
tripartite <- get_tripartite (data=tripartite_example)

tripartite$nodes$level1
tripartite$nodes$level2
tripartite$level1$level2$mat
tripartite
```

---

tripartite_example *An example of dataframe encoding a small tripartite graph.*

Description

`tripartite_example` is a `data.frame` containing raw data that encodes a small tripartite graph. It has the proper format to be loaded by `get_tripartite`. It contains four columns. Each row gives a link between two nodes belonging to two different levels of the tripartite graph. The first column gives the level of the first node (any integer in {1, 2, 3}) and the second column gives its name (any character string). Similarly, the third and fourth columns give the level and the name of the second node. A fifth column could eventually be added to give the weights of the links (any positive integer or float value).

Usage

```
data(tripartite_example)
```

Format

An object of class `data.frame` with 11 rows and 4 columns.

Examples

```
data (tripartite_example)
head (tripartite_example)

# Load the data.frame into a proper data structure
tripartite <- get_tripartite (data=tripartite_example)

# Get the names of the nodes in the second level of the tripartite graph
tripartite$nodes$level2

# Get the transition matrix of a random walk going from the level 2 to level 1
tripartite$transitions$level2$level1$mat
```

`triversity` *Compute diversity measures on tripartite graphs.*

Description

`triversity` is an R package for the computation of diversity measures on tripartite graphs. First, it implements a parametrized family of such diversity measures which apply on probability distributions. Sometimes called "True Diversity", this family contains famous measures such as the richness, the Shannon entropy, the Herfindahl-Hirschman index, and the Berger-Parker index. Second, the package allows to apply these measures on probability distributions resulting from random walks between the levels of tripartite graphs. By defining an initial distribution at a given level of the graph and a path to follow between the three levels, the probability of the walker's position within the final level is then computed, thus providing a particular instance of diversity to measure.

This package has been developed by researchers of the Complex Networks team, located within the Computer Science Laboratory of Paris 6 (LIP6), for the AlgoDiv project founded by the French National Agency of Research (ANR) under grant ANR-15-CE38-0001.

Links:

- AlgoDiv project: http://algodiv.huma-num.fr/
- Complex Networks team: http://www.complexnetworks.fr/
- LIP6: https://www.lip6.fr/
- ANR: http://www.agence-nationale-recherche.fr/

Contact:

- Robin Lamarche-Perrin: <Robin.Lamarche-Perrin@lip6.fr>
  See also my webpage: https://www-complexnetworks.lip6.fr/~lamarche/

List of main collaborators:

- Robin Lamarche-Perrin (CNRS, ISC-PIF, LIP6)

- Lionel Tabourier (UPMC, LIP6)
- Fabien Tarissan (CNRS, ISP, LIP6)
- Raphaël Fournier S'niehotta (CNAM, CÉDRIC)
- Rémy Cazabet (UdL, LIRIS)

Index