Package ‘FCMapper’

February 19, 2015

Type Package

Title Fuzzy Cognitive Mapping

Version 1.0

Date 2014-12-16

Author Shaun Turney and Michael Bachhofer

Maintainer Shaun Turney <shaun.turney@mail.mcgill.ca>

Description Provides several functions to create and manipulate fuzzy cognitive maps. It is based on FCMap-per for Excel, distributed at http://www.fcmappers.net/joomla/, developed by Michael Bachhofer and Martin Wildenberg. Maps are inputted as adjacency matrices. Attributes of the maps and the equilibrium values of the concepts (including with user-defined constrained values) can be calculated. The maps can be graphed with a function that calls ‘igraph’. Multiple maps with shared concepts can be aggregated.

Imports igraph

License GPL-2

NeedsCompilation no

Repository CRAN

Date/Publication 2015-02-06 01:00:43

R topics documented:

FCMapper-package ................................................................. 2
changes.scenario ................................................................. 3
check.matrix ....................................................................... 4
combine.maps ................................................................. 5
comp.maps ................................................................. 6
comp.scenarios ................................................................. 7
concept.indices ................................................................. 8
graph.fcm ................................................................. 9
matrix.indices ................................................................. 10
nochanges.scenario ............................................................. 11

Index 13
Description

This package provides several functions to create and manipulate fuzzy cognitive maps. It is based on FCMapper for Excel, distributed at http://www.fcmappers.net/joomla/, developed by Michael Bachhofer and Martin Wildenberg.

Maps are inputted as adjacency matrices. Attributes of the maps and the equilibrium values of the concepts (including with user-defined constrained values) can be calculated. The maps can be graphed with a function that calls "igraph". Multiple maps with shared concepts can be aggregated.

Details

<table>
<thead>
<tr>
<th>Package</th>
<th>FCMapper</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
<td>Package</td>
</tr>
<tr>
<td>Version</td>
<td>1.0</td>
</tr>
<tr>
<td>Date</td>
<td>2014-12-16</td>
</tr>
<tr>
<td>License</td>
<td>GPL-2</td>
</tr>
</tbody>
</table>

Functions:
- `changes.scenario`: Finds equilibrium values of concepts with no fixed values.
- `check.matrix`: Checks that fuzzy cognitive map is square and values are between -1 and 1.
- `combine.maps`: Aggregates multiple fuzzy cognitive maps into one.
- `comp.scenarios`: Compares the equilibrium values of two scenarios.
- `comp.maps`: Calculates similarity indices (S2 and Jaccard) of two cognitive maps.
- `concept.indices`: Calculates concept-level indices.
- `graph.fcm`: Plots fuzzy cognitive map by calling igraph package
- `matrix.indices`: Calculates matrix-level indices.
- `nochanges.scenario`: Finds equilibrium values of concepts with user-defined fixed values.

Author(s)

Shaun Turney and Michael Bachhofer
Maintainer: Shaun Turney <shaun.turney@mail.mcgill.ca>

References

http://www.fcmappers.net/
changes.scenario


---

changes.scenario  Scenario with fixed values

Description

The equilibrium values of the concepts in a fuzzy cognitive map are calculated. One or more concept values can be fixed by the user to be within 0 or 1. At each iteration (time step) the fixed concepts are set back to their chosen values. A plot of the concept values over the iterations is given.

Usage

changes.scenario(matrix, concept.names, iter, set.concepts, set.values)

Arguments

matrix  A quantitative fuzzy cognitive map.
concept.names  A quantitative character vector.
iter  Number of iterations.
set.concepts  A character vector.
set.values  A numeric vector.

Details

The fuzzy cognitive map should be in the form of quantitative adjacency matrices. The concept.names input is the names of the concepts in the fuzzy cognitive map. set.concepts is a character vector of the concepts the user wishes to fix, while set.values gives the desired values in the same order.

Value

A dataframe containing the equilibrium values of the concepts. If equilibrium has not been reached a warning will be printed. A plot of the concept values over the iterations is given.

Author(s)

Shaun Turney

See Also

nochanges.scenario
Examples

```r
matrix = matrix(nrow=7,ncol=7)
matrix[1,] = c(0,-.5,0,0,1,0,1)
matrix[2,] = c(1,0,1,0,2,0,0,0.6)
matrix[3,] = c(0,1,0,0,0,0,0)
matrix[4,] = c(0.6,0,0,1,0,0,0.1)
matrix[5,] = c(0,0.5,0,0,1,0,-0.6)
matrix[6,] = c(0,0,-1,0,0,0,0)
matrix[7,] = c(0,0,0,-0.5,0,0,1)
concept.names = c("A", "B", "C", "D", "E", "F", "G")

changes.scenario(matrix,concept.names,iter=25,set.concepts=c("B", "G"),set.values=c(0.5,0))
```

Description

The fuzzy cognitive map is checked to confirm that it is square and all values are within -1 and 1.

Usage

```
check.matrix(matrix)
```

Arguments

- `matrix` A quantitative fuzzy cognitive matrix.

Details

The fuzzy cognitive map should be in the form of a quantitative adjacency matrix.

Value

If the matrix is not square then a warning will be printed. If the matrix contains values not between -1 and 1 then a warning will be printed along with the identify of the offending values. If the matrix is square or has no values outside of -1 and 1 then a confirmation is printed. The function checks the diagonal to determine whether there are self-loops and gives a warning if there are.

Author(s)

Shaun Turney
Examples

# Is not square and contains values outside of -1 and 1:
matrix = matrix(nrow=7,ncol=7)
matrix[1,] = c(0,-0.5,0,0,1,0,1)
matrix[2,] = c(1,0,1,0.2,0,0,0.6)
matrix[3,] = c(0,1,0,0,2,0)
matrix[4,] = c(0.6,0,0,1,0,0,0.1)
matrix[5,] = c(-5,0.5,0,0,1,0,-0.6)
matrix[6,] = c(0,0,-1,0,0,0,0)

# Check
check.matrix(matrix)

# Matrix without issues:
matrix2 = matrix(nrow=7,ncol=7)
matrix2[1,] = c(0,-0.5,0,0,1,0,1)
matrix2[2,] = c(1,0,1,0.2,0,0,0.6)
matrix2[3,] = c(0,1,0,0,2,0)
matrix2[4,] = c(0.6,0,0,1,0,0,0.1)
matrix2[5,] = c(0.5,0,0,1,0,-0.6)
matrix2[6,] = c(0,0,-1,0,0,0)
matrix2[7,] = c(0,0,0,-0.5,0,0,1)

# Check
check.matrix(matrix2)

---

**combine.maps**  
*Map aggregator*

**Description**

Aggregates two fuzzy cognitive maps into one.

**Usage**

`combine.maps(matrix1, matrix2, concept.names1, concept.names2)`

**Arguments**

- `matrix1` A quantitative fuzzy cognitive map.
- `matrix2` A quantitative fuzzy cognitive map.
- `concept.names1` A character vector.
- `concept.names2` A character vector.
Details
The fuzzy cognitive maps should be in the form of quantitative adjacency matrices. The concept.names inputs are the names of the concepts in the first and second fuzzy cognitive maps, respectively.

Value
An aggregated fuzzy cognitive map in adjacency matrix format. It contains the edges and concepts of the two inputted fuzzy cognitive maps. Edges which are shared between the two fuzzy cognitive maps are averaged in the aggregated map.

Author(s)
Shaun Turney

Examples
```
#Matrix 1
matrix = matrix(nrow=7,ncol=7)
matrix[1,] = c(0,-0.5,0,0,1,0,1)
matrix[2,] = c(1,0,1,0,2,0,0,0,0)
matrix[3,] = c(0,1,0,0,0,0,0,0)
matrix[4,] = c(0,0,0,1,0,0,0)
matrix[5,] = c(0,0.5,0,1,0,0,0,0,0,0)
matrix[6,] = c(0,0,1,0,0,0,0,0,0)
matrix[7,] = c(0,0,0,0,0,0,0,0,0,0,0,0,0,0,0)
concept.names = c("A","B","C","D","E","F","G")

#Matrix 2
matrix2 = matrix(nrow=6,ncol=6)
matrix2[1,] = c(0,1,0,1,0,0)
matrix2[2,] = c(-1,1,0,-1,0,1)
matrix2[3,] = c(0,0,1,0,0,1)
matrix2[4,] = c(-1,0,0,0,0,0)
matrix2[5,] = c(0,0.5,0,1,0,0)
matrix2[6,] = c(1,0,0,-1,0,0)
concept.names2 = c("E","F","G","H","I","J")

#Aggregate
combine.maps(matrix1=matrix,matrix2=matrix2,concept.names1=
concept.names,concept.names2=concept.names2)
```

Description
Calculates similarity indices (S2 and Jaccard) of two cognitive maps.
comp.scenarios

Usage

    comp.maps(concept.names1, concept.names2)

Arguments

    concept.names1  Character vector of concept names.
    concept.names2  Character vector of concept names.

Value

    The S2 and Jaccard similarity indices are calculated. S2 is the proportion of concepts that are
    shared between the two concept maps. Jaccard is calculated as a/(a+b+c) where "a is the number
    of concepts shared by the two maps, "b is the number of concepts present only in Map 1, and "c" is
    the number of concepts present only in Map 2.

Author(s)

    Shaun Turney

See Also

    changes.scenario, nochanges.scenario

Examples

    concept.names1 = c("A", "B", "C", "D", "E", "F", "G")
    concept.names2 = c("C", "D", "E", "F", "G", "H")

    comp.maps(concept.names1, concept.names2)

comp.scenarios  Scenario comparison

Description

    The equilibrium values of two scenarios are compared.

Usage

    comp.scenarios(scenario1, scenario2)

Arguments

    scenario1  Dataframe output from nochanges.scenario or changes.scenario.
    scenario2  Dataframe output from nochanges.scenario or changes.scenario.
Value
The difference between the equilibrium values of the two scenarios are given in a data frame along with the concept names, the scenario 1 values, the scenario 2 values, and the percent difference.

Note
The scenarios must be scenarios of fuzzy cognitive maps with the same concepts, though not necessarily the same edges.

Author(s)
Shaun Turney

See Also
changes.scenario nochanges.scenario

Examples
```r
matrix = matrix(nrow=7,ncol=7)
matrix[1,] = c(0,-0.5,0,0,1,0,1)
matrix[2,] = c(1,0,1,0.2,0,0,0.6)
matrix[3,] = c(0,1,0,0,0,0,0)
matrix[4,] = c(0.6,0,0,1,0,0,0.1)
matrix[5,] = c(0,0.5,0,0,1,0,-0.6)
matrix[6,] = c(0,0,-1,0,0,0,0)
matrix[7,] = c(0,0,0,-0.5,0,0,1)
concept.names = c("A","B","C","D","E","F","G")

scenario1 = nochanges.scenario(matrix,iter=25,concept.names)
scenario2 = changes.scenario(matrix,concept.names,iter=25,set.concepts=c("B"),set.values=c(0.5))
compscenario(scenario1,scenario2)
```

## Concept indices

<table>
<thead>
<tr>
<th>concept.indices</th>
<th>Concept indices</th>
</tr>
</thead>
</table>

Description
Concept-level indices are calculated, including the out-degree, in-degree, centrality, and whether it is a transmitter, receiver, ordinary or unconnected concept.

Usage
```r
concept.indices(matrix, concept.names)
```
Arguments

- **matrix**: A quantitative fuzzy cognitive map.
- **concept.names**: A character vector.

Details

The fuzzy cognitive map should be in the form of a quantitative adjacency matrix. The concept.names input is the names of the concepts in the fuzzy cognitive map.

Value

A dataframe containing the concept name, out-degree, in-degree, centrality, and whether it is a transmitter, receiver, ordinary or unconnected concept.

Author(s)

Shaun Turney

Examples

```r
matrix = matrix(nrow=7,ncol=7)
matrix[1,] = c(0,-0.5,0,0,1,0,1)
matrix[2,] = c(1,0,1,0.2,0,0,0.6)
matrix[3,] = c(0,1,0,0,0,0,0)
matrix[4,] = c(0.6,0,1,0,0,0.1)
matrix[5,] = c(0,0.5,0,1,0,-0.6)
matrix[6,] = c(0,0,-1,0,0,0)
matrix[7,] = c(0,0,0,-0.5,0,0,1)
concept.names = c("A", "B", "C", "D", "E", "F", "G")

concept.indices(matrix, concept.names)
```

Description

Creates a visual representation (circles and arrows) of the fuzzy cognitive map by calling igraph. The thickness of the arrows is determined by the magnitude of the edge value. Negative edges are red while positive edges are black. The size of the circles (concepts) is defined by the user.

Usage

```r
graph.fcm(matrix, concept.sizes, concept.names)
```
### matrix.indices

#### Arguments

- **matrix**: A quantitative fuzzy cognitive map.
- **concept.sizes**: A numeric vector the same length as the number of concepts.
- **concept.names**: A character vector.

#### Details

The fuzzy cognitive maps should be in the form of quantitative adjacency matrices. The `concept.names` input is the names of the concepts in the fuzzy cognitive map. The `concept.sizes` is a vector of values between 0 and 1 and determines the size of the circles. The user may want to define `concept.sizes` as the equilibrium values of the concepts.

#### Value

An igraph plot with circles representing concepts and arrows representing edges.

#### Author(s)

Shaun Turney

#### Examples

```r
matrix = matrix(nrow=7,ncol=7)
matrix[1,] = c(0,-0.5,0,0,1,0,1)
matrix[2,] = c(1,0,1,0.2,0,0,0.6)
matrix[3,] = c(0,1,0,0,0,0)
matrix[4,] = c(0.6,0,0.1,0,0,0.1)
matrix[5,] = c(0,0.5,0,1,0,-0.6)
matrix[6,] = c(0,0,-1,0,0,0)
matrix[7,] = c(0,0,-0.5,0,0,1)
concept.names = c("A","B","C","D","E","F","G")

results = nochanges.scenario(matrix,iter=25,concept.names)

graph.fcm(matrix,concept.sizes=results$Equilibrium_value,concept.names)
```

---

### matrix.indices

**Matrix indices**

#### Description

Matrix-level indices are calculated, including the number of connections, connection density, number of concepts, number of transmitters, number of receivers, number of no connections, number of ordinary, number of self-loops, connections per variable, complexity, and hierarchy.
Usage

matrix.indices(matrix)

Arguments

matrix  A quantitative fuzzy cognitive matrix.

Details

The fuzzy cognitive maps should be in the form of a quantitative adjacency matrix.

Value

A dataframe containing the number of connections, connection density, number of concepts, number of transmitters, number of receivers, number of no connections, number of ordinary, number of self-loops, connections per variable, complexity, and hierarchy.

Author(s)

Shaun Turney

References


Examples

matrix = matrix(nrow=7,ncol=7)
matrix[1,] = c(0,-0.5,0,0,1,0,1)
matrix[2,] = c(1,0.1,0.2,0,0,0.6)
matrix[3,] = c(0,1,0,0,0,0)
matrix[4,] = c(0.6,0,1,0,0,0.1)
matrix[5,] = c(0,0.5,0,1,0,-0.6)
matrix[6,] = c(0,0,-1,0,0,0)
matrix[7,] = c(0,0,0,-0.5,0,1)
concept.names = c("A","B","C","D","E","F","G")
matrix.indices(matrix)

nochanges.scenario

Scenario with no fixed values

Description

The equilibrium values of the concepts in a fuzzy cognitive map are calculated. A plot of the concept values over the iterations is given. It activates the matrix with a vector of 1s and squeezes the resulting vector with a logic function. The function checks for convergence and gives a warning if convergence isn’t reached. A plot of the concept values over the iterations is given.
Usage

nochanges.scenario(matrix, concept.names, iter)

Arguments

matrix A quantitative fuzzy cognitive map.
concept.names A character vector.
iter The number of iterations.

Details

The fuzzy cognitive map should be in the form of quantitative adjacency matrices. The concept.names input is the names of the concepts in the fuzzy cognitive map.

Value

A dataframe containing the equilibrium values of the concepts. If equilibrium has not been reached a warning will be printed. A plot of the concept values over the iterations is given.

Author(s)

Shaun Turney

See Also

nochanges.scenario

Examples

```r
matrix = matrix(nrow=7,ncol=7)
matrix[1,] = c(0,-0.5,0,0,1,0,1)
matrix[2,] = c(1,0,1,0.2,0,0,0.6)
matrix[3,] = c(0,1,0,0,0,0,0)
matrix[4,] = c(0.6,0,0,1,0,0,0.1)
matrix[5,] = c(0,0.5,0,0,1,0,-0.6)
matrix[6,] = c(0,0,-1,0,0,0,0)
matrix[7,] = c(0,0,0,-0.5,0,0,1)
concept.names = c("A","B","C","D","E","F","G")

nochanges.scenario(matrix,iter=25,concept.names)
```
Index

*Topic package
   FCMapper-package, 2
changes.scenario, 2, 3, 7, 8
check.matrix, 2, 4
combine.maps, 2, 5
comp.maps, 2, 6
comp.scenarios, 2, 7
concept.indices, 2, 8

FCMapper (FCMapper-package), 2
FCMapper-package, 2

graph.fcm, 2, 9

matrix.indices, 2, 10

nochanges.scenario, 2, 3, 7, 8, 11, 12