Package 'rcausim'

June 2, 2024

Title Generate Causally-Simulated Data

Version 0.0.1

Description Generate causally-simulated data to serve as ground truth for evaluating methods in causal discovery and effect estimation. The package provides tools to assist in defining functions based on specified edges, and conversely, defining edges based on functions. It enables the generation of data according to these predefined functions and causal structures. This is particularly useful for researchers in fields such as artificial intelligence, statistics, biology, medicine, epidemiology, economics, and social sciences, who are developing a general or a domain-specific methods to discover causal structures and estimate causal effects. Data simulation adheres to principles of structural causal modeling.

Depends R (>= 4.4.0)

Imports dplyr,
ggplot2,
magrittr,
purrr,
readr,
stringr,
tibble,
tidyr

Suggests broom,
ggnetwork,
ggpubr,
igraph,
kableExtra,
knitr,
rmarkdown,
testthat

VignetteBuilder knitr

License GPL-3

Encoding UTF-8

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.1

Contents

`data_from_function` *Generate causally-simulated data*

Description

Generate causally-simulated data

Usage

```
data_from_function(func, n)
```

Arguments

`func`	Functions, an object class generated by `function_from_edge` or `function_from_user` functions. The causal structure needs to be a directed acyclic graph (DAG), which means no loops are allowed. Use `edge_from_function` to identify edges given a list of functions, then draw a causal diagram using the edges data frame (see vignettes). At least a function in the list must include 'n' as the only argument. All arguments within any function must be defined by their respective functions, except the argument 'n'. The output lengths of vertex functions must match the specified length 'n'.
`n`	Number of observations, a numeric of length 1, non-negative, and non-decimal.

Value

A data frame which include the simulated data for each vertex as a column.

Examples

```
data(functions)
data_from_function(functions, n = 100)
```

define *Define a function in the list of functions*

Description

Define a function in the list of functions

Usage

```
define(func, which, what)
```

Arguments

func	Functions, an object class generated by function_from_edge or function_from_user functions.
which	Which, a character of length 1 indicating a vertex name for which function is defined. The vertex name must be defined in 'Functions'.
what	What, a function to be defined. It must use all and only the specified arguments for the vertex in 'Functions'.

Value

A list of either functions or character vectors of arguments for function. It can be continuously defined or redefined by a user using define function. If all elements of the list are functions, then it can be an input for generating the simulated data.

Examples

```
data(edges)
functions <- function_from_edge(edges)
function_B <- function(n){ rnorm(n, 90, 5) }
functions <- define(functions, 'B', function_B)
```

edges *Edges*

Description

An example of a data frame which include the columns 'from' and 'to in this order. A vertex name 'n' does not exist.

Usage

```
edges
```

Format

A data frame with 7 rows and 2 columns:

from A vertex name from which a directed edge comes.

tp A vertex name to which a directed edge comes.

Source

This package

---

edge_from_function *Identify edges given functions*

Description

Identify edges given functions

Usage

```
edge_from_function(func)
```

Arguments

func	Functions, an object class generated by function_from_edge or function_from_user functions.

Value

A data frame which include the columns 'from' and 'to in this order.

Examples

```
data(functions)
edge_from_function(functions)
```

---

functions *Functions*

Description

An example of an object class generated by function_from_edge or function_from_user functions. The causal structure is a directed acyclic graph (DAG), which means no loops are allowed. A function in the list include 'n' as the only argument. All arguments within any function are defined by their respective functions, except the argument 'n'. The output lengths of vertex functions match the specified length 'n'.

Usage

```
functions
```

Format

A list with 5 elements:

B A function with an argument 'n'.

A A function with an argument 'B'.

D A function with an argument 'A'.

C A function with arguments 'A', 'B', and 'D'.

E A function with arguments 'A' and 'C'.

Source

This package

`function_from_edge` *List functions given edges*

Description

List functions given edges

Usage

```
function_from_edge(e)
```

Arguments

`e`	Edge, a data frame that must only include the columns 'from' and 'to in this order. A vertex name 'n' is not allowed.

Value

A list of character vectors of arguments for function which will be defined by a user using `define` function.

Examples

```
data(edges)
function_from_edge(edges)
```

`function_from_user` *List functions from user*

Description

List functions from user

Usage

```
function_from_user(func)
```

Arguments

`func`	Functions, a list of functions which are defined by a user. The list must be non-empty. All elements of the list must be named. All elements of the list must be functions.

Value

A list of functions. It can be an input for generating the simulated data, or redefined by a user using `define` function.

Examples

```
function_B <- function(n){ rnorm(n, mean = 90, sd = 5) }
function_A <- function(B){ ifelse(B>=95, 1, 0) }
functions <- list(A = function_A, B = function_B)
functions <- function_from_user(functions)
```

`print.Functions` *Print method for Functions*

Description

Print method for Functions

Usage

```
## S3 method for class 'Functions'
print(x)
```

Arguments

`func`	Functions, an object class generated by `function_from_edge` or `function_from_user` functions.

Value

A summary of vertices that has functions. If there are vertices without functions, an instruction is shown.

Examples

```
data(edges)
functions <- function_from_edge(edges)
print(functions)
```

`time_varying` *Generate time-varying data*

Description

Generate time-varying data

Usage

```
time_varying(func, data, T_max)
```

Arguments

func	Functions, an object class generated by function_from_edge or function_from_user functions. The causal structure needs to be a directed cyclic graph (DCG), which means loops are allowed. Use edge_from_function to identify edges given a list of functions, then draw a causal diagram using the edges data frame (see vignettes). All arguments within any function must be defined by their respective functions, except the argument 'n'. The output lengths of vertex functions must match the 'data' row number.
data	Data, a data frame generated by data_from_function which contains causally-simulated data at t=0. Column names of 'i', 't', and 't_max' are not allowed, which respectively refer to instance, time, and maximum time.
T_max	Maximum time for every instance, a numeric vector of length equal to the number of rows in 'data' and must be non-negative and non-decimal.

Value

A data frame which include the simulated data for each vertex as a column for each time up to maximum time for every instance.

Examples

```
data(functions)
simulated_data <- data_from_function(functions, n = 100)

function_B <- function(B){
  B + 1
}

functions <- define(functions, which = "B", what = function_B)
T_max <- rpois(nrow(simulated_data), lambda = 25)

time_varying(functions, data = simulated_data, T_max = T_max)
```