Package 'MetaboLouise'

November 30, 2018

Version 1.0.0

Date 2018-11-20

Title METABOlomics LOngitudinal SimUlation Engine

Author Charlie Beirnaert

Maintainer Charlie Beirnaert <charlie_beirnaert@icloud.com>

Description

Simulating dynamic (longitudinal, time-resolved) metabolomics data based on an underlying biological network. The network is initiating with certain concentrations and evolves over a simulated time period. Optionally external influxes (concentration drivers) can be added.

Depends R (>= 3.1.0),

Imports igraph, graphics, stats

Suggests knitr, rmarkdown

LazyData true

VignetteBuilder knitr

License GPL-3

RoxygenNote 6.1.0

NeedsCompilation no

Repository CRAN

Date/Publication 2018-11-30 16:50:13 UTC

R topics documented:

DataSimulateR *DataSimulateR*

Description

Simulated dynamic/longitudinal data based on an underlying network. The network is initialized with values for every node (e.g. concentrations in the case of metabolites). These values evolve over time caused by the (variable) rates.

Usage

```
DataSimulateR(NetworkMatrix, dT, Tstart, Tstop, T0_nodes = 100,
  influx_vector = NULL, influx_Tframe = NULL, rate_vector = NULL,
  rate_mapping = NULL, RateFunctionObject = NULL, plot_out = TRUE,
  plot_title = NULL)
```

Arguments

NetworkMatrix	The underlying network in matrix form (e.g. form the NetworkCreateR function).
dT	Simulation time step (must be small enough to avoid approximation errors).
Tstart	Starting time point.
Tstop	Ending time point.
T0_nodes	Vector (or single value) with the initial node value(s) (metabolite concentrations).
influx_vector	Vector with the influx (per time unit) received by the corresponding metabolites.
influx_Tframe	Vector of two values indicating the start and ending time of the influx (if only single ending time value supplied, assumption of influx start = Tstart is made). This can also be a data frame/matrix with 2 columns and 1 row per metabolite, or just a single ending value.
rate_vector	Vector with the initial rates. Number of rates can be between 1 and the number of edges in the network.
rate_mapping	A matrix (same size as NetworkMatrix) with for every link/edge in the network (1 in NetworkMatrix) an index of rate_vector to be matched.
RateFunctionObject	An object from RateFunctionBuildR describing the evolution of rates. If not provided, the rates are constant.
plot_out	Whether to plot the simulated data.
plot_title	Optional plot title.

Value

A list with: the time vector and a matrix with the simulated data. (1 row per node)

Author(s)

Charlie Beirnaert, <charlie.beirnaert@uantwerpen.be>

Examples

```
Nmetabos <- 20L
Nrates <- 10L

Network <- NetworkCreateR(N = Nmetabos, BA_power = 0.5, BA_mValue = 4)

Rate_function <- RateFunctionBuildR(type = "sigmoid")

rate_vector <- round(5*runif(Nrates))

rate_mapping <- Network
active_rates <- which(Network == 1, arr.ind = TRUE)
for(rr in 1:nrow(active_rates)){
   rate_mapping[active_rates[rr,1], active_rates[rr,2]] <- sample(seq_along(rate_vector), size = 1)
    }

 No_influx <- DataSimulateR(NetworkMatrix = Network, dT = 0.01, Tstart = 0, Tstop = 3,
                      T0_nodes = 100, rate_vector = rate_vector, rate_mapping = rate_mapping,
                           RateFunctionObject = Rate_function, plot_out = TRUE)
```

GetFoldChanges *GetFoldChanges*

Description

Calculate the fold change equivalent for the final state of two simulated datasets e.g. with vs without influx.

Usage

```
GetFoldChanges(ReferenceDataObject, AlternativeDataObject,
  plot_out = TRUE, bw = 0.05, plot_title = NULL)
```

Arguments

ReferenceDataObject	The reference data object (result from DataSimulateR function)
AlternativeDataObject	The alternative situation data object (result from DataSimulateR function)
plot_out	Whether to plot the Fold change distribution.
bw	The bw for the density plot.
plot_title	Optional plot title.

Value

A list with: the time vector and a matrix with the simulated data. (1 row per node)

Author(s)

Charlie Beirnaert, <charlie.beirnaert@uantwerpen.be>

Examples

```
Nmetabos <- 20L
Nrates <- 10L

Network <- NetworkCreateR(N = Nmetabos, BA_power = 0.5, BA_mValue = 4)

Rate_function <- RateFunctionBuildR(type = "sigmoid")

rate_vector <- round(5*runif(Nrates))

rate_mapping <- Network
active_rates <- which(Network == 1, arr.ind = TRUE)
for(rr in 1:nrow(active_rates)){
   rate_mapping[active_rates[rr,1], active_rates[rr,2]] <- sample(seq_along(rate_vector), size = 1)
    }

 No_influx <- DataSimulateR(NetworkMatrix = Network, dT = 0.01, Tstart = 0, Tstop = 3,
                      T0_nodes = 100, rate_vector = rate_vector, rate_mapping = rate_mapping,
                           RateFunctionObject = Rate_function, plot_out = FALSE)

 influx_vector <- c(rep(1,10),rep(0,Nmetabos-10))
 With_influx <- DataSimulateR(NetworkMatrix = Network, dT = 0.01, Tstart = 0, Tstop = 3,
                        T0_nodes = 100, influx_vector = influx_vector, influx_Tframe = 0.5,
                              rate_vector = rate_vector, rate_mapping = rate_mapping,
                              RateFunctionObject = Rate_function, plot_out = FALSE)

GetFoldChanges(ReferenceDataObject = No_influx, AlternativeDataObject = With_influx)
```

NetworkCreateR *NetworkCreateR*

Description

This function creates a fictional metabolomics network according to the Barabasi-Albert model.

Usage

```
NetworkCreateR(N, BA_power = 0.5, BA_mValue = 4, plothist = TRUE,
  histbreaks = 50, ...)
```

Arguments

N	the number of nodes in the network (metabolites)
BA_power	(igrap parameter) The power of the preferential attachment.
BA_mValue	(igrap parameter) The number of edges to add in each time step
plothist	Whether to plot a histogram of the network's connectivity distribution
histbreaks	The number of breaks in the histogram
...	Optional additional arguments to be passed along to the network generator function sample_pa.

Value

A matrix with the network structure

Author(s)

Charlie Beirnaert, <charlie.beirnaert@uantwerpen.be>

Examples

```
Network <- NetworkCreateR(N = 50, BA_power = 0.5, BA_mValue = 4)
image(Network)
```

RateFunctionBuildR *RateFunctionBuildR*

Description

Build and visualize an appropriate rate multiplier function. The flow between nodes in the network is governed by certain rates. These rates can be made dependent on the values of the source node (flow from source node to receiver node). With this function you can visualize different types and receive the parameters to be submitted to the simulation function. Note that this function returns an object containing the plot parameters. The simulation function can take only function type.

Usage

```
RateFunctionBuildR(type = c("linear", "sigmoid", "step"),
  C_range = c(0, 200), lin_xy_start = c(0, 0), lin_slope = 0.01,
  sig_C0 = 100, sig_k = 0.05, sig_max = 2, step_levels = c(0, 1,
  2), step_switchpoints = c(50, 150), plot.out = TRUE,
  Nplotpoints = 100)
```

Arguments

type	Type of the function, any or multiple of "linear", "sigmoid" or "step" can be used.
C_range	The range of node concentrations over which to plot the rate function.
lin_xy_start	Linear parameter: the starting point for the linear curve e.g. c(0,0).
lin_slope	Linear parameter: In case only a single point is provided, the slope is also necessary.
sig_C0	Sigmoid parameter: The midpoint concentration.
sig_k	Sigmoid parameter: The curve steepness.
sig_max	Sigmoid parameter: The maximal height of the function.
step_levels	Step function parameter: the distinct levels of the function.
step_switchpoints	Step function parameter: The points (nr. of levels minus 1) at which a switch is made.
plot.out	Whether to plot the resulting functions.
Nplotpoints	The number of plotpoints for the optional plot

Value

A plot with the visualize rate function and a list with the necessary parameters

Author(s)

Charlie Beirnaert, <charlie.beirnaert@uantwerpen.be>

Examples

```
RateFunctionBuildR()
```

Index