Package ‘Cascade’

March 24, 2014

Type  Package

Title  Selection, Reverse-Engineering and Prediction in Cascade networks.

Version  1.03

Date  2014-03-24

Author  Nicolas Jung, Frederic Bertrand, Myriam Maumy-Bertrand Laurent Vallat

Maintainer  <cascade@math.unistra.fr>

Description  The Cascade is a modeling tool allowing gene selection, reverse engineering, and prediction.

License  GPL (>= 2)

Depends  methods, abind, animation, cluster, graphics, grDevices, grid, igraph, lars, lattice, limma, magic, methods, nnls, splines, stats, stats4, survival, tnet, utils, VGAM


R topics documented:

  Cascade-package .......................................................... 1
  analyze_network .......................................................... 2
  as.micro_array .......................................................... 3
  compare ................................................................. 4
  cutoff ................................................................. 5
  dim ................................................................. 6
  evolution ............................................................... 6
  geneNeighborhood ....................................................... 7
  genePeakSelection ....................................................... 8
  gene_expr_simulation ..................................................... 9
  head ............................................................... 10
  inference ............................................................ 10
  micropredict-class ..................................................... 11
  micro_array-class ..................................................... 12
## The Cascade Package

### Description

The Cascade is a modeling tool allowing gene selection, reverse engineering, and prediction.

### Details

<table>
<thead>
<tr>
<th>Package</th>
<th>Cascade</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
<td>Package</td>
</tr>
<tr>
<td>Version</td>
<td>1.03</td>
</tr>
<tr>
<td>Date</td>
<td>2014-03-24</td>
</tr>
<tr>
<td>License</td>
<td>GNU 2.0</td>
</tr>
<tr>
<td>Depends</td>
<td>methods</td>
</tr>
</tbody>
</table>

### Author(s)

This package has been written by Nicolas Jung in collaboration with Frederic Bertrand, Myriam Maumy-Bertrand and Laurent Vallat. Maintainer: <cascade@math.unistra.fr>

### References


analyze_network

Analyse the network

Description
Calculates some indicators for each node in the network.

Usage
analyze_network(Omega, nv, ...)

Arguments
Omega  a network object
nv     the level of cutoff at which the analysis should be done
...    label_v : (optionnal) name of the genes

Value
A matrix containing, for each node, its betweenness, its degree, its output, its closeness.

Author(s)
Nicolas Jung, Frédéric Bertrand, Myriam Maumy-Bertrand.

References


Examples
data(network)
analyze_network(network, nv=0)
Description

Coerce a matrix into a micro_array object.

Usage

```r
as.micro_array(M, time, subject)
```

Arguments

- `M` A matrix. Contains the microarray measurements. Should of size $N \times K$, with $N$ the number of genes and $K=T \times P$ with $T$ the number of time points, and $P$ the number of individuals. This matrix should be created using `cbind(M1,M2,...)` with $M1$ a $N \times T$ matrix with the measurements for individual 1, $M2$ a $N \times T$ matrix with the measurements for individual 2.
- `time` A vector. The time points measurements.
- `subject` The number of subjects.

Value

A micro_array object.

Author(s)

Nicolas Jung, Frédéric Bertrand, Myriam Maumy-Bertrand.

References


Examples

```r
data(micro_US)
micro_US<-as.micro_array(micro_US,time=c(60,90,210,390),subject=6)
```
**compare**

Some basic criteria of comparison between actual and inferred network.

Description

Allows comparison between actual and inferred network.

Usage

```r
compare(Net, Net_inf, nv)
```

Arguments

- **Net**: A network object containing the actual network.
- **Net_inf**: A network object containing the inferred network.
- **nv**: A number that indicates at which level of cutoff the comparison should be done.

Value

A vector containing: sensibility, predictive positive value, and the F-score

Author(s)

Nicolas Jung, Frédéric Bertrand, Myriam Maumy-Bertrand.

References


**cutoff**

Choose the best cutoff

Description

Allows estimating the best cutoff, in function of the scale-freeness of the network. For a sequence of cutoff, the corresponding p-value is then calculated.

Usage

```r
cutoff(Omega, ...)
```
Arguments

Omega a network object
... Optional arguments:
sequence a vector corresponding to the sequence of cutoffs that will be tested.
x_min an integer; only values over x_min are further retained for performing the test.

Value

A list containing two objects:

- p.value the p values corresponding to the sequence of cutoff
- p.value.inter the smoothed p value vector, using the loess function

Author(s)

Nicolas Jung, Frédéric Bertrand, Myriam Maumy-Bertrand.

References


Examples

data(network)
cutoff(network)
#See vignette for more details

---

**dim**

*Dimension of the data*

**Description**

Dimension of the data

**Methods**

signature(x = "micro_array") Gives the dimension of the matrix of measurements.
evolution

See the evolution of the network with change of cutoff

Description

See the evolution of the network with change of cutoff. This function may be useful to see if the global topology is changed while increasing the cutoff.

Usage

evolution(net,list_nv,...)

Arguments

net       a network object
list_nv   a vector of cutoff at which the network should be shown
...       Optionnal arguments:
gr       a vector giving the group of each gene
color.vertex a vector giving the color of each node
fix       logical, should the position of the node in the network be calculated once at the beginning? Default to TRUE.
taille    vector giving the size of the plot. Default to c(2000,1000)
...       see plot function

Value

A HTML page with the evolution of the network.

Author(s)

Nicolas Jung, Frédéric Bertrand, Myriam Maumy-Bertrand.

References


Examples

data(network)
sequence<-seq(0,0.2,length.out=20)
evolution(network,sequence)
geneNeighborhood  

*Find the neighborhood of a set of nodes.*

**Description**

Find the neighborhood of a set of nodes.

**Usage**

```
geneNeighborhood(net, targets, ...)```

**Arguments**

- `net`  
  a network object
- `targets`  
  a vector containing the set of nodes
- `...`  
  Optional arguments. See plot options.

**Value**

The neighborhood of the targeted genes.

**Author(s)**

Nicolas Jung, Frédéric Bertrand, Myriam Maumy-Bertrand.

**References**


**Examples**

#See vignette
Methods for selecting genes

Description

Selection of differentially expressed genes.

Usage

geneSelection(x, y, tot.number, ...)
genePeakSelection(x, pic, ...)

Arguments

x
either a micro_array object or a list of micro_array objects. In the first case, the micro_array object represents the stimulated measurements. In the second case, the control unstimulated data (if present) should be the first element of the list.

y
either a micro_array object or a list of strings. In the first case, the micro_array object represents the stimulated measurements. In the second case, the list is the way to specify the contrast:

First element: condition, condition&time or pattern. The condition specification is used when the overall is to compare two conditions. The condition&time specification is used when comparing two conditions at two precise time points. The pattern specification allows to decide which time point should be differentially expressed.

Second element: a vector of length 2. The two conditions which should be compared. If a condition is used as control, it should be the first element of the vector. However, if this control is not measured throughout time, the option cont=TRUE should be used.

Third element: depends on the first element. It is no needed if condition has been specified. If condition&time has been specified, then this is a vector containing the time point at which the comparison should be done. If pattern has been specified, then this is a vector of 0 and 1 of length T, where T is the number of time points. The time points with desired differential expression are provided with 1.

tot.number
an integer. The number of selected genes. If tot.number <0 all differentially genes are selected. If tot.number > 1, tot.number is the maximum of differentially genes that will be selected. If 0<tot.number<1, tot.number represents the proportion of differentially genes that are selected.

pic
integer. At which time points measurements should the genes be selected [optionnal for geneSelection].

... Optional arguments:

M2 a micro_array object. The unstimulated measurements.

data_log logical (default to TRUE) : should data be logged ?

wanted.patterns a matrix with wanted patterns [only for geneSelection].
**gene_expr_simulation**

*forbidden.patterns* a matrix with forbidden patterns [only for geneSelection].

*durPic* vector of size 2 (default to c(1,1)) : the first elements gives the length of the pic at the left, the second at the right. [only for genePeakSelection]

*abs_val* logical (default to TRUE) : should genes be selected on the basis of their absolute value expression ? [only for genePeakSelection]

*alpha_diff* float ; the risk level

**Value**

A micro_array object.

**Author(s)**

Nicolas Jung, Frédéric Bertrand, Myriam Maumy-Bertrand.

**References**


```
gene_expr_simulation

Simulates microarray data based on a given network.
```

**Description**

Simulates microarray data based on a given network.

**Usage**

```r
gene_expr_simulation(network, ...)
```

**Arguments**

- `network` A network object.
- `...` `time_label` a vector containing the time labels.
- `subject` the number of subjects
- `level_pic` the mean level of pics.

**Value**

A micro_array object.
head

Author(s)

Nicolas Jung, Frédéric Bertrand, Myriam Maumy-Bertrand.

References


Examples

#See vignette

head

Overview of a micro_array object

Description

Overview of a micro_array object.

Methods

signature(x = "ANY") Gives an overview.
signature(x = "micro_array") Gives an overview.

inference

Reverse-engineer the network

Description

Reverse-engineer the network.

Usage

inference(M,...)
micropredict-class

Arguments

\[ M \]
a micro_array object.

... Optional arguments:

\[ \text{tour.max}=30 \] maximal number of steps.
\[ g=\text{function}(x) \ 1/x \] the new solution is chosen as (the old solution + g(x) * the new solution)/(1+g(x)) where x is the number of steps.
\[ \text{conv}=10\text{e}-3 \] convergence criterion.
\[ \text{cv.subjects}=\text{TRUE} \] should the cross validation be done removing the subject one by one ?
\[ \text{nb.folds}=\text{NULL} \] Relevant only if \text{cv.subjects} is FALSE. The number of folds in cross validation.
\[ \text{eps}=10\text{e}-5 \] machine zero
\[ \text{type.inf}=\text{"iterative"} \] “iterative” or “noniterative” : should the algorithm be computed iteratively

Value

A network object.

Author(s)

Nicolas Jung, Frédéric Bertrand, Myriam Maumy-Bertrand.

References


Examples

#data(micro_US)
#inference(micro_US)
#See vignette for more details

micropredict-class  Class "micropred"
micro_array-class

Objects from the Class

Objects can be created by calls of the form new("micropred", ...).

Examples

showClass("network")

---

micro_array-class  Class "micro_array"

Description

The Class

Objects from the Class

Objects can be created by calls of the form new("micro_array", ...).

Slots

- microarray: Object of class "matrix" ~~
- name: Object of class "vector" ~~
- group: Object of class "vector" ~~
- start_time: Object of class "vector" ~~
- time: Object of class "vector" ~~
- subject: Object of class "numeric" ~~

Methods

dim signature(x = "micro_array"): ...
genePicSelection signature(M1 = "micro_array", M2 = "micro_array", pic = "numeric"): ...
  ...
geneSelection signature(x = "micro_array", y = "micro_array", tot.number = "numeric"): ...
  ...
geneSelection signature(x = "list", y = "list", tot.number = "numeric"): ...
  ...
head signature(x = "micro_array"): ...
inference signature(M = "micro_array"): ...
plot signature(x = "micro_array", y = "ANY"): ...
plot signature(x = "micro_array", y = "ANY"): ...
plot signature(x = "micropredict", y = "ANY"): ...
predict signature(object = "micro_array"): ...
print signature(x = "micro_array"): ...
summary signature(object = "micro_array"): ...
unionMicro signature(M1 = "micro_array", M2 = "micro_array"): ...
Examples

showClass("micro_array")

---

**micro_S**

*Stimulated data*

**Description**

Stimulated data

**Usage**

data(micro_S)

**Format**

A data frame with 54613 probesets measured 6 times through 4 time points.

**References**


**Examples**

data(micro_S)

---

**micro_US**

*Unstimulated data*

**Description**

Unstimulated data

**Usage**

data(micro_US)

**Format**

A data frame with 54613 probesets measured 6 times through 4 time points.
References


Examples

```r
data(micro_US)
```

network  

A network object data.

Description

A network object (the same as the result in the vignette).

Usage

```r
data(network)
```

Examples

```r
data(network)
plot(network)
print(network)
```

network-class  

Class "network"

Description

2254

Objects from the Class

Objects can be created by calls of the form `new("network", ...).`
Slots

- network: Object of class "matrix"
- name: Object of class "vector"
- F: Object of class "array"
- convF: Object of class "matrix"
- convO: Object of class "vector"
- time_pt: Object of class "vector"

Methods

- analyze_network signature(Omega = "network")
- cutoff signature(Omega = "network")
- evolution signature(net = "network")
- geneNeighborhood signature(net = "network")
- plot signature(x = "network", y = "ANY")
- plot signature(x = "network", y = "micro_array")
- position signature(net = "network")
- print signature(x = "network")

Examples

showClass("network")

network_random Generates a network.

Description

Generates a network.

Usage

network_random(nb, time_label, exp, init, regul, min_expr, max_expr, casc.level)

Arguments

- nb Integer. The number of genes.
- time_label Vector. The time points measurements.
- exp The exponential parameter, as in the barabasi.game function in igraph package.
- init The attractiveness of the vertices with no adjacent edges. See barabasi.game function.
- regul A vector mapping each gene with its number of regulators.
- min_expr Minimum of strength of a non-zero link
- max_expr Maximum of strength of a non-zero link
- casc.level ...
Value

A network object.

Author(s)

Nicolas Jung, Frédéric Bertrand, Myriam Maumy-Bertrand.

References


plot-methods

Plot

Description

Considering the class of the argument which is passed to plot, the graphical output differs.

Methods

```r
signature(x = "micro_array", y = "ANY", ...) x a micro_array object
  list_nv a vector of cutoff at which the network should be shown

signature(x = "network", y = "ANY", ...) x a network object
  ... Optionnal arguments:
    gr a vector giving the group of each gene
    choice what graphic should be plotted: either "F" (for a representation of the matrices F) or "network".
    nv the level of cutoff. Default to 0.
    ini using the “position” function, you can fix the position of the nodes
    color.vertex a vector defining the color of the vertex
    ani vector giving the size of the plot. Default to c(2000,1000)
    video if ani is TRUE and video is TRUE, the animation result is a GIF video
    label_v vector defining the vertex labels
    legend.position position of the legend
    frame.color color of the frames
    label.hub logical : if TRUE only the hubs are labeled
    edge.arrowsize size of the arrows ; default to 0.7
    edge.thickness edge thickness ; default to 1.

signature(x = "micropredict", y = "ANY", ...) x a micropredict object
  ... Optionnal arguments: see plot for network
```
**position-methods**  

*Returns the position of edges in the network*

**Description**  

Returns the position of edges in the network

**Methods**  

```r
signature(net = "network")  
Returns a matrix with the position of the node. This matrix can then be used as an argument in the plot function.
```

**predict**  

*Prediction of the gene expressions after a knock-out experience*

**Description**  

Prediction of the gene expressions after a knock-out experience

**Usage**  

```r
predict(object,...)
```

**Arguments**  

- `object`  
a micro_array object
- `...`  
Other arguments:
  - `Omega`  
a network object.
  - `nv`  
[=0] numeric ; the level of the cutoff
  - `targets`  
[NULL] vector ; which genes are knocked out ?

**Author(s)**  

Nicolas Jung, Frédéric Bertrand, Myriam Maumy-Bertrand.

---

**print-methods**  

*~ Methods for Function print ~*

**Description**  

Methods for function `print`
Methods for Function `summary`

**Description**

Methods for function `summary`

**unionMicro-methods**  
Makes the union between two `micro_array` objects.

**Description**

Makes the union between two `micro_array` objects.

**Methods**

- `signature(M1 = "micro_array", M2 = "micro_array")`  
  Returns a `micro_array` object which is the union of M1 and M2.

- `signature(M1 = "list", M2 = "ANY")`  
  Returns a `micro_array` object which is the union of the elements of M1.