Package ‘RTCA’

March 26, 2015

Type Package
Title Open-source toolkit to analyse data from xCELLigence System (RTCA)
Version 1.19.0
Date 2012-01-11
Author Jitao David Zhang
Maintainer Jitao David Zhang <davidvonpku@gmail.com>
Description Import, analyze and visualize data from Roche(R) xCELLigence RTCA systems. The package imports real-time cell electrical impedance data into R. As an alternative to commercial software shipped along the system, the Bioconductor package RTCA provides several unique transformation (normalization) strategies and various visualization tools.
License LGPL-3
LazyLoad yes
Depends methods,stats,graphics,Biobase,RColorBrewer, gtools
Suggests xtable
biocViews CellBasedAssays, Infrastructure, Visualization, TimeCourse
NeedsCompilation no

R topics documented:
alphaNames .................................................. 2
combineRTCA .................................................. 3
controlView .................................................. 4
derivativeTransform .......................................... 6
factor2numeric ................................................ 7
interpolationTransform ....................................... 8
nearestTimeIndex ............................................. 9
## alphaNames

Auxilliary functions for experiments with microtitre plates

### Description

Functions to manipulate indices or names of microtitre plates

### Usage

```r
alphaNames(row = 8, column = 12, order=c("column","row"))
repairAlphaName(x)
alphaNames2Pos(x)
rowcol2pos(row = 1, column=1, plateFormat=c("96","384"))
```

### Arguments

- **row**: integer, row index, 1,...,8 for 96-well plates
- **column**: integer, column index, 1,...,12 for 96-well plates
- **x**: character, Well alpha name, in the form of [A-Z][0-9][0-9], like 'A01'
- **order**: character, should the alpha names returned in a row-first or column-first order?
- **plateFormat**: integer, the microtitre format, either 96 or 384

### Details

- **alphaNames** returns so-called *alpha well names* in the form of [A-H][0-9][0-9] (i.e., A01, C03, D11, H12) for microtitre plates. The order of returned alphaNames is controlled by the option `order`, which can be set either as `col` or `row`.
- **repairAlphaName** attempts to fix incomplete alpha well names. Now it is mainly used to fix well names missing the leading 0 of numeric index, like 'A1'.
- **alphaName2Pos** returns the row and column number of the given alpha well name, in the form of two-column data frame with `row` and `col` as colnames.
- **rowcol2pos** returns the row-wise position index of given row and column index.
combineRTCA

Value
See details

Author(s)
Jitao David Zhang <jitao_david.zhang@roche.com>

Examples

```r
wells <- alphaNames()
repairAlphaName("A1")
alphaNames2Pos(c("A01","B02","C03","H12"))
rowcol2pos(3,1)
```

combineRTCA  Combine a list of RTCA objects

Description
Combine a list of RTCA objects

Usage
```r
combineRTCA(list)
```

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>list</td>
<td>A list of RTCA objects</td>
</tr>
</tbody>
</table>

Details

The current implementation requires all the objects have exactly the same time-points recorded (or at least of same length).

The combined RTCA object has an obligatory column in the phenoData ‘Plate’ (upper-case!), which matches the names of the RTCA list. When the list has no names, the ‘Plate’ field is filled with integer index starting from 1.

Value
A new RTCA object

Note
Special attention should be given to the cases where the list parameter partially has names. In this case all items without name will be assigned to a ‘Plate’ field of empty string ("""). Therefore it is advised either to assign names to all items of the list, or leave them all off.
controlView

PLOT CONTROL WELLS IN RTCA DATA

Description

A convenience function to plot sample wells with control wells on an E-plate in RTCA system. To use the function the phenoData field of the RTCA object must contain a field named “GeneSymbol”.

Usage

controlView(rtca, genesymbol = c("Allstar", "COPB2", "GFP", "mock", "PLK1", "WEE1"), cols, ylim, smooth, group, interval, drawsd, normline, ncol, legendpos, pdataNcolumn)

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>rtca</td>
<td>An object of RTCA. To use the function, the phenoData must contain a column which name is specified by the pData.column parameter.</td>
</tr>
<tr>
<td>genesymbol</td>
<td>character, gene symbols to be plotted.</td>
</tr>
<tr>
<td>cols</td>
<td>character, colors used by the provided gene symbols</td>
</tr>
<tr>
<td>ylim</td>
<td>y-axis lim</td>
</tr>
<tr>
<td>smooth</td>
<td>logical, whether the RTCA object should be smoothed before plotting</td>
</tr>
<tr>
<td>group</td>
<td>logical. If ‘group’ is set to TRUE, wells with the same GeneSymbol will be summarized and plotted. For instance, these could be biological replicates. Otherwise each well is plotted separately</td>
</tr>
</tbody>
</table>
controlView

```r
ylab y axis label
xlab x axis label
drawsd logical, should the error bar be drawn to represent standard deviation?
normline logical, should the base-time indicated by a line? See ratioTransform for the concept of the base-time
ncol integer, legend column number
legendpos character, legend position
pDataNcolumn The column which the genesymbol parameter will be matched with
... other parameters passed to the plot function
```

Details
The function is often called to draw sample and control in one plot.

Value
NULL, the function is called for its side effect

Author(s)
Jitao David Zhang <jitao_david.zhang@roche.com>

See Also
RTCA

Examples
```r
require(RTCA)

ofile <- system.file("extdata/testOutput.csv", package="RTCA")
pfile <- system.file("extdata/testOutputPhenoData.csv", package="RTCA")

pData <- read.csv(pfile, sep="\t", row.names="Well")
metaData <- data.frame(labelDescription=c("Rack number",
"siRNA catalogue number",
"siRNA gene symbol",
"siRNA EntrezGene ID",
"siRNA targeting accession"))

phData <- new("AnnotatedDataFrame", data=pData, varMetadata=metaData)
x <- parseRTCA(ofile, phenoData=phData)

controlView(x, genesymbol=c("mock","COPB2","PLK1"), ylim=c(0,2))
```
derivativeTransform  DERIVATIVE TRANSFORM OF RTCA OBJECT

Description

Derivative transform of RTCA object, returning the change rate of cell impedance

Usage

derivativeTransform(object)

Arguments

object  An object of RTCA

Details

The first derivative of the cell impedance curve measured by RTCA. The derivative of the last time point is estimated by that of the next to last point.

Value

An RTCA object populated with derivative values

Author(s)

Jitao David Zhang <jitao_david.zhang@roche.com>

See Also

smoothTransform and interpolationTransform for smoothing and interpolating the RTCA data. rgrTransform calculates relative growth rate, which calls derivativeTransform.

Examples

require(RTCA)
ofile <- system.file("/extdata/testOutput.csv", package="RTCA")x <- parseRTCA(ofile)
xDeriv <- derivativeTransform(x)
Description

The functions implement easy interface to certain tasks of factor. See details for explanation.

Usage

factor2numeric(x)
relevels(x, refs)

Arguments

x  A vector of factor
refs  A vector of character, reference vector to give the order of levels

Details

relevels re-arrange the order of levels by the given character refs. Alternatively user could use factor(..., levels=refs) to achieve a similar effect, however the relevels enables also partial list. The missing levels in refs will be ordered to the last.

factor2numeric converts factor of numerics into their numeric form.

Value

A vector of factor

Author(s)

Jitao David Zhang <jitao_david.zhang@roche.com>

Examples

```r
## factor2numeric
numFac <- factor(c(3.5, 2.5, 2.5, 3.5, 1))
numFac
levels(numFac)

factor2numeric(numFac)
class(factor2numeric(numFac))

## relevels
relevels(numFac, c("3.5", "1", "2.5"))
relevels(numFac, c("3.5", "2.5"))
```
interpolationTransform

TRANSFORM RTCA DATA WITH INTERPOLATION

Description

Interpolate RTCA data

Usage

interpolationTransform(object, interval=0.01, method=c("linear","constant","fmm","periodic","natural"))

Arguments

object An RTCA object
... other parameters, interval and method are implemented, see below
interval numeric, the interval between interpolated points, set to 0.01 by default
method character, specifying the method for interpolation, “linear” by default (for linear interpolation). Allowed options are: “linear” and “constant” for approx interpolation, and “fmm”, “periodic”, “natural” and “monoH.FC” for cubic spline interpolation

Details

Since most RTCA experiments record the experiments in the irregular time-series, sometimes however it is desired to have regular intervals. interpolationTransform interpolate between data points to estimate results of regular intervals.

Two classes of interpolations are supported by now: linear (using approx) and cubic spline (spline) interpolation. By default linear interpolation is used.

Value

An interpolated object of RTCA.

Author(s)

Jitao David Zhang <jitao_david.zhang@roche.com>

See Also

rgrTransform stands for relative growth rate transformation, ratioTransform for ratio normalization adopted by Roche commercial software, smoothTransform to smooth the RTCA readout.
**nearestTimeIndex**

Examples

```r
require(RTCA)

ofile <- system.file("/extdata/testOutput.csv", package="RTCA")
x <- parseRTCA(ofile)
xInter <- interpolationTransform(x)
```

---

**nearestTimeIndex**  
*GET INDEX FOR NEAREST TIME*

Description

Get index for the nearest time point to the given one. Called internally in many time-point related functions.

Usage

```r
nearestTimeIndex(rtca, time)
```

Arguments

- `rtca`  
  An object of `RTCA`
- `time`  
  numeric, a time point

Details

The function finds the time point with minimum absolute difference to the given time and returns its index.

Value

An integer, the index of the nearest time point

Author(s)

Jitao David Zhang <jita_david.zhang@roche.com>

See Also

- [timepoints](https://example.com) to return all time points of an `RTCA` object.
Examples

```
require(RTCA)

ofile <- system.file("/extdata/testOutput.csv", package="RTCA")
x <- parseRTCA(ofile)

x
xIndex <- nearestTimeIndex(x, 25)
timepoints(x)[xIndex]
```

Description

The function parses RTCA output file into RTCA object

Usage

```
parseRTCA(file, dec = ".", phenoData, maskWell, ...)
```

Arguments

- **file**: character, name of the RTCA output file
- **dec**: decimal sign of the file
- **phenoData**: phenoData
- **maskWell**: character, either names or regular expression pattern(s) for well(s) to mask
- **...**: other parameters passed to `read.table`

Details

A csv-like format file can be exported from the RTCA device, which can be fed into this function to set up an instance of RTCA object.

In the `/extdata/` directory of the package, such a file is provided as an example. The first line contains the experiment ID, which is followed by a matrix of recorded data in the tabular form. The first and second column records the time-interval in the unit of hour and hour-minute-second format respectively. The rest columns then record the read-out (‘Cell-Index’, or ‘CI’) of the device, with each well a role.

`phenoData` allows user to annotate the wells. Its usage mimicks the `ExpressionSet` object in the `biobase` package.

`maskWell` allows to mask wells in case, for example, they are known to be contaminated. The values can be either a vector of well names, or a regular expression pattern for wells to be masked. To learn regular expression patterns see `grep`.
**Value**

An object of RTCA-class

**Author(s)**

Jitao David Zhang <jitao_david.zhang@roche.com>

**References**

http://www.roche-applied-science.com/proddata/gpip/3_8_9_1_1_1.html

**Examples**

```r
require(RTCA)

ofile <- system.file("extdata/testOutput.csv", package="RTCA")
pfile <- system.file("extdata/testOutputPhenoData.csv", package="RTCA")

pData <- read.csv(pfile, sep="\t", row.names="Well")
metaData <- data.frame(labelDescription=c("Rack number","siRNA catalogue number","siRNA gene symbol","siRNA EntrezGene ID","siRNA targeting accession"))

phData <- new("AnnotatedDataFrame", data=pData, varMetadata=metaData)
x <- parseRTCA(ofile, phenoData=phData)

print(x)

## mask wells, e.g. due to unusual values
x.skip <- parseRTCA(ofile, phenoData=phData, maskWell=c("D09"))
x.skip.multiWells <- parseRTCA(ofile, phenoData=phData, maskWell=c("A01", "B01", "C02"))

## skip the last row
x.skip.pattern <- parseRTCA(ofile, phenoData=phData, maskWell=c("H[0-9]{2}"))

## check the number of masked wells
noMasked <- function(x) sum(apply(x, 2, function(x) all(is.na(x))))
noMasked(exprs(x.skip))
noMasked(exprs(x.skip.multiWells))
noMasked(exprs(x.skip.pattern))
```
**Description**

Plots a *E-plate* in RTCA assays in one plot to convey an overview of the plate.

**Usage**

```
plateView(rtca, ylim, titles,...)
```

**Arguments**

- `rtca` An object of `RTCA`
- `ylim` ylab lim
- `titles` Titles of sub-figures representing each well. If missing, the function seeks whether a *Well* column is available in the pData of the RTCA object, and if so, its value will be used. If not, the sample names (by `sampleNames` function) will be used as titles.
- `...` Other parameters passed to the `plot` function. Currently options `col`, `lty` and `lwd` are supported. See details below.

**Details**

For now the function only supports the visualization of a 96-well *E-plate*.

The plate view plot draws lines indicating cell index (or its transformations) in a birdview. When `...` are not specified, default color, line style and width are used. `col`, `lty` and `lwd` can be a vector, and if needed they will be expanded to have the same length as wells.

**Value**

`NULL`, the function is called for the side effect

**Author(s)**

Jitao David Zhang <jitao_david.zhang@roche.com>

**See Also**

`RTCA` for data structure, `plot` for the basic plot function.
Examples

```
require(RTCA)

ofile <- system.file("extdata/testOutput.csv", package="RTCA")
rtca <- parseRTCA(ofile)

## Not run automatically, because of 'margin too large'
## plateView(rtca)
## plateView(rtca, lty=2)
## plateView(rtca, col=rep(1:8, each=12))

rtca.skip <- parseRTCA(ofile, maskWell="H[0-9](2)")
## plateView(rtca.skip)
```

---

### plotGridEffect

**PLOT GRID EFFECT OF RTCA**

**Description**

Plot the mean and deviation of rows/columns of a RTCA E-plate, to provide hints of potential row/column effect of the plate

**Usage**

```
plotGridEffect(rtca, mode = c("column", "row"), xlab = "time point", ylab = "readout", legend = TRUE, col, ...)
```

**Arguments**

- `rtca` An object of RTCA
- `mode` character, either "column" or "row", to choose which effect to depict
- `xlab` x-axis label
- `ylab` y-axis label
- `legend` logical, whether the legend should be added
- `col` Color of the curves
- `...` Further parameters passed to `plot` function

**Details**

The error bars depicts the standard deviations

**Value**

`NULL`, the function is called for its side effect
Author(s)
Jitao David Zhang

Examples

```
require(RTCA)

ofile <- system.file("extdata/testOutput.csv", package="RTCA")
x <- parseRTCA(ofile)
plotGridEffect(x)
```

rationTransform RATIO TRANSFORMATION OF RTCA DATA

Description

Performs ratio transformation (normalisation) of RTCA data, as recommended by the producer Roche.

Usage

```
rationTransform(object, time)
```

Arguments

- `object` An object of RTCA
- `time` numeric, the time point used to normalize the whole series of data

Details

The xCelligence software provided by Roche performs ratio transform implicitly by dividing the time-series impedance measurement by the value of a selected time point (so-called 'base-time'), for instance 5 hours after compound transfection, in each cell. The aim of this transformation was to scale (normalize) the data of different wells, since the normalized values of all wells are uniformly 1 at the base-time.

However, this method is vulnerable to arbitrary selection of the time point chosen to normalize. It may be helpful to try several base-time values before comparing normalized results.

See `derivativeTransform` and `rgrTransform` for other normalization (scaling) possibilities.

Value

An object of RTCA, populated with normalized value. The normalized values of all wells are uniformly 1 at the base-time.

Author(s)

Jitao David Zhang <jitao_david.zhang@roche.com>
**rgrTransform**

**See Also**

*smoothTransform* and *interpolationTransform* for smoothing and interpolating the RTCA data. *rgrTransform* calculates relative growth rate, *derivativeTransform* calculates derivative. The later two methods are not sensitive to the selection of base-time point.

**Examples**

```
require(RTCA)

ofile <- system.file("/extdata/testOutput.csv", package="RTCA")
x <- parseRTCA(ofile)

xNorm <- ratioTransform(x, 35)
```

---

**Description**

Transform RTCA data into relative growth rate

**Usage**

```
rgrTransform(object, smooth)
```

**Arguments**

- **object**: An object of *RTCA*
- **smooth**: logical, should the object be smooth transformed after the *rgrTransform*? Set to TRUE by default

**Details**

TODO: relative growth rate

**Value**

An object of *RTCA* populated with relative growth rate instead of input data

**Author(s)**

Jitao David Zhang <jitao_david.zhang@roche.com>

**References**

TODO: reference
RTCA-class

See Also
dervativeTransform for first derivative. ratioTransform for ratio normalization adopted by Roche commercial software. smoothTransform and interpolationTransform for other transformation possibilities.

Examples

require(RTCA)

ofile <- system.file("/extdata/testOutput.csv", package="RTCA")
x <- parseRTCA(ofile)

xRgr <- rgrTransform(x)

---

RTCA-class  Class "RTCA"

Description

RTCA object

Objects from the Class

Objects can be created by calls of the form new("RTCA", assayData, phenoData, featureData, experimentData, annotation).

However, it is more common to be constructed by parseRTCA function by reading in RTCA output data directly.

Slots

expID: Object of class "character", experiment ID
timeline: Object of class "RTCAtimeline", recording action track along the time line
assayData: Object of class "AssayData", assay data inherited from ExpressionSet-class
phenoData: Object of class "AnnotatedDataFrame", pheno data of the assay, annotating the wells
featureData: Object of class "AnnotatedDataFrame", feature data of the assay, preserved for time-line recording by the package
experimentData: Object of class "MIAME", idle
annotation: Object of class "character", idle
.__classVersion__: Object of class "Versions", idle

Extends

Class ExpressionSet-class, directly. Class eSet-class, by class "ExpressionSet", distance 2. Class VersionedBiobase-class, by class "ExpressionSet", distance 3. Class Versioned-class, by class "ExpressionSet", distance 4.
RTCA-class

Methods

**addAction** signature(`object = "RTCA", time = "numeric", action = "character")**: add action at the specified time, passed to the RTCA timeline slot

**getAction** signature(`object = "RTCA", time = "numeric")**: get action at the specified time, passed to the RTCA timeline slot

**plotRTCA** signature(`x = "RTCA")**: plot RTCA

**rmAction** signature(`object = "RTCA", time = "numeric")**: remove action at the specified time, passed to the RTCA timeline slot

**show** signature(`object = "RTCA")**: print method

**expID** codesignature(`object = "RTCA")**: get Experiment ID

**expID<-** codesignature(`object = "RTCA", value = "ANY")**: set Experiment ID

**time** signature(`x = "RTCA")**: deprecated

**timeline** signature(`object = "RTCA")**: get the RTCA timeline slot

**timeline<-** signature(`object = "RTCA")**: assign the RTCA timeline slot

**timepoints** signature(`object = "RTCA")**: get the recording time points in a vector

**timepoints<-** signature(`object = "RTCA")**: assign the recording time points

**updateAction** signature(`object = "RTCA", time = "numeric", action = "character")**: update the action at the specified time, passed to the RTCA timeline slot

**plot** signature(`x = "RTCA", y)**: plot the RTCA running plot with `matplot`. `y` is interpreted as the indices of the columns to be plotted, and will be expanded to all the columns in case it is missing.

Author(s)

Jitao David Zhang <jitao_david.zhang@roche.com>

References

1 https://www.roche-applied-science.com/sis/xcelligence/index.jsp?id=xect_000000 introduces xCelligence system.

2 http://www.roche-applied-science.com/proddata/gpip/3_8_9_1_1_1.html for brief introduction into RTCA

Examples

`new("RTCA", expID="testExp01")`
Description

Time line of actions performed by the xCelligence device, supporting CRUD manipulations (create, read, update and delete).

Objects from the Class

Objects can be created by calls of the form `new("RTCAtimeline")`. However, it is more common to be called implicitly by creating an instance of RTCA object.

Slots

- actionTrack: Object of class "data.frame", records action track in the form of two-column data.frame. The two columns must have the names 'time' and 'action'.
- timeUnit: Object of class "character", recording the unit of time points stored in the actionTrack slot.
- startTime: Object of class "POSIXct", the absolute time when the measurement started (at the time point '0')

Methods

- `addAction` signature(object = "RTCAtimeline", time = "numeric", action = "character"):
  add action at the specified time
- `actionTrack` signature(object = "RTCAtimeline"): get the action track in the form of data.frame
- `actionTrack<-` signature(object = "RTCAtimeline", value = "data.frame"): assign the action track
- `getAction` signature(object = "RTCAtimeline", time = "numeric"): get action at the specified time
- `orderAction` signature(object = "RTCAtimeline"): order the action track by the time
- `reset` signature(object = "RTCAtimeline"): undo all editing of the object and reset it to the initial state
- `rmAction` signature(object = "RTCAtimeline", time = "numeric"): remove the action at the specified time
- `timeUnit` signature(object = "RTCAtimeline"): return the time unit used by the action track
- `timeUnit<-` signature(object = "RTCAtimeline", value = "character"): assign the time unit used by the action track
- `start` signature(object = "RTCAtimeline"): return the starting POSIXct time of the experiment
- `timeUnit<-` signature(object = "RTCAtimeline", value = "character"): assign the starting POSIXct time of the experiment
sliceRTCA

Author(s)

Jitao David Zhang <jitao_david.zhang@roche.com>

References

1 http://www.xcelligence.roche.com/ introduces xCelligence system.
2 http://www.roche-applied-science.com/proddata/gpip/3_8_9_1_1_1.html for brief introduction into RTCA

See Also

RTCA

Examples

tl <- new("RTCAtimeline")
show(tl)

---

sliceRTCA

SLICE RTCA OBJECT WITH TIME

Description

Subset (slice) RTCA object with starting- and ending-time

Usage

sliceRTCA(x, start, end)

Arguments

x An object of RTCA
start numeric, start time
end numeric, end time

Details

In case the exact starting- or ending-time is not matched, the nearest time point will be used to subset.

Value

An object of RTCA

Author(s)

Jitao David Zhang <jitao_david.zhang@roche.com>
Examples

```r
require(RTCA)

ofile <- system.file("/extdata/testOutput.csv", package="RTCA")
x <- parseRTCA(ofile)

subx <- sliceRTCA(x, 20, 50)
```

---

**smoothTransform**

**SMOOTH TRANSFORM OF RTCA OBJECT**

Description

Smoothing the RTCA cell impedance measurement

Usage

`smoothTransform(object, ...)`

Arguments

- `object` An object of `RTCA`
- `...` Parameters passed to `smooth.spline`

Details

`smoothTransform` smooths the RTCA cell impedance measurement by calling the function `smooth.spline`. This feature can be useful for visualization purposes and in conjunction with other transformations.

Value

An `RTCA` object populated with smoothed values

Note

`ratioTransform` performs ratio transformation recommended by the machine provider. `interpolationTransform` for interpolating the RTCA data. `derivativeTransform` returns cell impedance change rates and `rgrTransform` calculates relative growth rate.

Author(s)

Jitao David Zhang <jitao_david.zhang@roche.com>
spectramaxImport

Examples

```r
require(RTCA)
ofile <- system.file("/extdata/testOutput.csv", package="RTCA")
x <- parseRTCA(ofile)
xSmooth <- smoothTransform(x)
```

Description

Import output files from Spectramax spectrophotometer (plate reader) into the list format compatible with the cellHTS2 package.

Usage

```r
spectramaxImport(file, encoding="latin1")
```

Arguments

- `file`: A Spectramax file
- `encoding`: File character encoding, by default “latin1”

Details

The function imports output files from Spectramax plate reader, with which single-channel cell-based assays could be performed. Such assay includes WST-1 viability assay, which can be used to validate RTCA assay results.

Value

A list of two items: one data frame (no name) and one character vector (`txt`). The data frame contains following columns:

- `well`: Well indices ([A-Z][0-9][0-9] format) on the microtitre plate
- `val`: Value of each well

The character vector `txt` contains a copy of the file contents.

Author(s)

Jitao David Zhang <jitao_david.zhang@roche.com>

See Also

cellHTS2 package documentation.
Examples

```r
wstFiles <- dir(system.file("extdata", package="RTCA"),
    pattern="^WST.*csv$", full.names=TRUE)
spectramaxImport(wstFiles[1])

## NOT RUN
## spectramaxImport also supports multiple files, in which case the
## result is a list of individual lists
spectramaxImport(wstFiles)
## END NOT RUN
```
Index

*Topic IO
  parseRTCA, 10
*Topic classes
  RTCA-class, 16
  RTCAtimeline-class, 18
*Topic file
  parseRTCA, 10
*Topic hplot
  controlView, 4
  plateView, 12
  plotGridEffect, 13
*Topic misc
  factor2numeric, 7
*Topic models
  rgrTransform, 15
*Topic ts
  interpolationTransform, 8
  actionTrack (RTCAtimeline-class), 18
  actionTrack, RTCAtimeline-method (RTCAtimeline-class), 18
  actionTrack<- (RTCAtimeline-class), 18
  actionTrack<-, RTCAtimeline, data.frame-method (RTCAtimeline-class), 18
  addAction (RTCAtimeline-class), 18
  addAction, RTCA, numeric, character-method (RTCA-class), 16
  addAction, RTCAtimeline, numeric, character-method (RTCAtimeline-class), 18
  alphaNames, 2
  alphaNames2Pos (alphaNames), 2
  approx, 8
  combineRTCA, 3
  controlView, 4
  derivativeTransform, 6, 14, 16, 20
  derivativeTransform, RTCA-method (RTCA-class), 16
  expID (RTCA-class), 16
  expID, RTCA-method (RTCA-class), 16
  expID< (RTCA-class), 16
  expID<-, RTCA-method (RTCA-class), 16
  factor2numeric, 7, 7
  actionTrack (RTCAtimeline-class), 18
  actionTrack, RTCA, numeric-method (RTCA-class), 16
  actionTrack, RTCAtimeline, numeric-method (RTCAtimeline-class), 18
  grep, 10
  interpolationTransform, 6, 8, 15, 16, 20
  interpolationTransform, RTCA-method (RTCA-class), 16
  matplot, 17
  nearestTimeIndex, 9
  actionTrack (RTCAtimeline-class), 18
  actionTrack, RTCAtimeline-method (RTCAtimeline-class), 18
  actionTrack<- (RTCAtimeline-class), 18
  actionTrack<-, RTCAtimeline, data.frame-method (RTCAtimeline-class), 18
  addAction (RTCAtimeline-class), 18
  addAction, RTCA, numeric, character-method (RTCA-class), 16
  addAction, RTCAtimeline, numeric, character-method (RTCAtimeline-class), 18
  alphaNames, 2
  alphaNames2Pos (alphaNames), 2
  approx, 8
  combineRTCA, 3
  controlView, 4
  derivativeTransform, 6, 14, 16, 20
  derivativeTransform, RTCA-method (RTCA-class), 16
  expID (RTCA-class), 16
  expID, RTCA-method (RTCA-class), 16
  expID< (RTCA-class), 16
  expID<-, RTCA-method (RTCA-class), 16
  factor2numeric, 7, 7
  actionTrack (RTCAtimeline-class), 18
  actionTrack, RTCA, numeric-method (RTCA-class), 16
  actionTrack, RTCAtimeline, numeric-method (RTCAtimeline-class), 18
  grep, 10
  interpolationTransform, 6, 8, 15, 16, 20
  interpolationTransform, RTCA-method (RTCA-class), 16
  matplot, 17
  nearestTimeIndex, 9
  orderAction (RTCAtimeline-class), 18
  orderAction, RTCAtimeline-method (RTCAtimeline-class), 18
  parseRTCA, 10, 16
  plateView, 12
  plot, 5, 12, 13
  plot, RTCA-method (RTCA-class), 16
  plotGridEffect, 13
  plotRTCA, RTCA-method (RTCA-class), 16
  ratioTransform, 5, 8, 14, 16, 20
  ratioTransform, RTCA-method (RTCA-class), 16
  read.table, 10
  relevelevel, 7
  relevelevel (factor2numeric), 7
  repairAlphaName (alphaNames), 2
  reset (RTCAtimeline-class), 18
reset, RTCA timeline-method (RTCA timeline-class), 18
ggrTransform, 6, 8, 14, 15, 16, 20
ggrTransform, RTCA-method (RTCA-class), 16
rmAction (RTCA timeline-class), 18
rmAction, RTCA, numeric-method (RTCA-class), 16
rmAction, RTCA timeline, numeric-method (RTCA timeline-class), 18
rowcol2pos (alphaNames), 2
RTCA, 4–6, 8–10, 12–15, 18–20
RTCA-class, 16
RTCA timeline, RTCA-method (RTCA-class), 16
RTCA timeline-class, 18
RTCA timeline<-, RTCA-method (RTCA-class), 16

show, RTCA-method (RTCA-class), 16
sliceRTCA, 19
smooth.spline, 20
smoothTransform, 6, 8, 15, 16, 20
smoothTransform, RTCA, ANY-method (RTCA-class), 16
smoothTransform, RTCA, missing-method (RTCA-class), 16
spectramaxImport, 21
spline, 8
startTime (RTCA timeline-class), 18
startTime, RTCA timeline-method (RTCA timeline-class), 18
startTime<-, (RTCA timeline-class), 18
startTime<-, RTCA timeline, POSIXct-method (RTCA timeline-class), 18

time, RTCA-method (RTCA-class), 16
timeline (RTCA-class), 16
timeline, RTCA-method (RTCA-class), 16
timeline<-, (RTCA-class), 16
timeline<-, RTCA-method (RTCA-class), 16
timepoints, 9
timepoints (RTCA-class), 16
timepoints, RTCA-method (RTCA-class), 16
timepoints<-, (RTCA-class), 16
timepoints<-, RTCA-method (RTCA-class), 16
timeUnit (RTCA timeline-class), 18