

# Package ‘RTCA’

December 20, 2025

**Type** Package

**Title** Open-source toolkit to analyse data from xCELLigence System (RTCA)

**Version** 1.63.0

**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

**Collate** AllClasses.R AllGenerics.R AllMethods.R algorithms.R  
visualization.R RTCAfunctions.R spectramaxImport.R

**URL** <http://code.google.com/p/xcelligence/>,<http://www.xcelligence.roche.com/>,<http://www.nextbiomotif.com/Home/scientific-programming>

**biocViews** ImmunoOncology, CellBasedAssays, Infrastructure,  
Visualization, TimeCourse

**git\_url** <https://git.bioconductor.org/packages/RTCA>

**git\_branch** devel

**git\_last\_commit** d713410

**git\_last\_commit\_date** 2025-10-29

**Repository** Bioconductor 3.23

**Date/Publication** 2025-12-19

## Contents

alphaNames	2
combineRTCA	3
controlView	4
derivativeTransform	6
factor2numeric	7
interpolationTransform	8
nearestTimeIndex	9
parseRTCA	10
plateView	12
plotGridEffect	13
ratioTransform	14
rgrTransform	15
RTCA-class	16
RTCAtimeline-class	18
sliceRTCA	19
smoothTransform	20
spectramaxImport	21

Index	23
-------	----

---

alphaNames	<i>Auxilliary functions for experiments with microtitre plates</i>
------------	--

---

### Description

Functions to manipulate indices or names of microtitre plates

### Usage

```
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.

**Value**

See details

**Author(s)**

Jitao David Zhang <jitao\_david.zhang@roche.com>

**Examples**

```
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**

```
combineRTCA(list)
```

**Arguments**

list	A list of RTCA objects
------	------------------------

## 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.

## Author(s)

Jitao David Zhang <jitao\_david.zhang@roche.com>

## Examples

```
## An artificial example
require(RTCA)

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

xSub1 <- x[,1:3]
xSub2 <- x[,4:ncol(x)]
xComb <- combineRTCA(list(sub1=xSub1, sub2=xSub2))
identical(exprs(x), exprs(xComb))
pData(xComb)$Plate

## in case of nameless list
pData(combineRTCA(list(xSub1, xSub2)))$Plate

## partial names
pData(combineRTCA(list(a=xSub1, xSub2)))$Plate
```

## 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)
```

**Arguments**

rtca	An object of <a href="#">RTCA</a> .To use the function, the phenoData must contain a column which name is specified by the pData.column parameter.
genesymbol	character, gene symbols to be plotted.
cols	character, colors used by the provided gene symbols
ylim	y-axis lim
smooth	logical, whether the RTCA object should be smoothed before plotting
group	logical. If 'group' is set to TRUE, wells with the same <i>GeneSymbol</i> will be summarized and plotted. For instance, these could be biological replicates. Otherwise each well is plotted separately
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 <a href="#">ratioTransform</a> for the concept of the <i>base-time</i>
ncol	integer, legend column number
legendpos	character, legend position
pData.column	The column which the genesymbol parameter will be matched with
...	other parameters passed to the <a href="#">plot</a> 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

```
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.  
[rgtTransform](#) calculates relative growth rate, which calls [derivativeTransform](#).

**Examples**

```
require(RTCA)

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

xDeriv <- derivativeTransform(x)
```

---

**factor2numeric****FACTOR UTILITIES**

---

**Description**

The functions implement easy interface to certain tasks of factor. See [datails](#) for explaination

**Usage**

```
factor2numeric(x)
relevels(x, refs)
```

**Arguments**

<code>x</code>	A vector of factor
<code>refs</code>	A vector of character, reference vector to give the orderof 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

```
## 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, <code>interval</code> and <code>method</code> 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.

**Examples**

```
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**

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

**Arguments**

rtca	An object of <a href="#">RTCA</a>
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 <jitao\_david.zhang@roche.com>

**See Also**

[timepoints](#) 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]
```

---

parseRTCA

*PARSE RTCA OUTPUT FILE*

---

**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 <a href="#">read.table</a>

**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](http://www.roche-applied-science.com/proddata/gpip/3_8_9_1_1_1.html)

**Examples**

```
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))
noMasked(exprs(x.skip))
noMasked(exprs(x.skip.multiWells))
noMasked(exprs(x.skip.pattern))
```

---

**plateView****PLATE VIEW OF RTCA DATA**

---

**Description**

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

**Usage**

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

**Arguments**

<code>rtca</code>	An object of <a href="#">RTCA</a>
<code>ylim</code>	ylab lim
<code>titles</code>	Titles of sub-figures representing each well. If missing, the function seeks whether a <i>Well</i> column is available in the <code>pData</code> of the RTCA object, and if so, its value will be used. If not, the sample names (by <code>sampleNames</code> function) will be used as titles.
<code>...</code>	Other parameters passed to the <a href="#">plot</a> function. Currently options <code>col</code> , <code>lty</code> and <code>lwd</code> 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 <a href="#">RTCA</a>
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 <a href="#">plot</a> 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)
```

**ratioTransform**

*RATIO TRANSFORMATION OF RTCA DATA*

**Description**

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

**Usage**

```
ratioTransform(object, time)
```

**Arguments**

<b>object</b>	An object of <a href="#">RTCA</a>
<b>time</b>	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>

## 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)
```

---

rgrTransform

*TRANSFORM RTCA DATA INTO RELATIVE GROWTH RATE*

---

## Description

Transform RTCA data into relative growth rate

## Usage

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

## Arguments

object	An object of <a href="#">RTCA</a>
smooth	logical, should the object be smooth transformed after the <a href="#">rgrTransform</a> ? 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

## See Also

[derivativeTransform](#) 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, exprs, ...)`. 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 "RTCA timeline", 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.

## Methods

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

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

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

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

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

**plot** signature(x = "RTCA", y): plot the RTCA running plot with `matplotlib`. 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](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](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")
```

---

 RTCAtimeline-class      *Class "RTCA timeline"*


---

### 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("RTCA timeline")`. 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 = "RTCA timeline"`, `time = "numeric"`, `action = "character"`): add action at the specified time

**actionTrack** signature(`object = "RTCA timeline"`): get the action track in the form of `data.frame`

**actionTrack<-** signature(`object = "RTCA timeline"`, `value = "data.frame"`): assign the action track

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

**orderAction** signature(`object = "RTCA timeline"`): order the action track by the time

**reset** signature(`object = "RTCA timeline"`): undo all editing of the object and reset it to the initial state

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

**timeUnit** signature(`object = "RTCA timeline"`): return the time unit used by the action track

**timeUnit<-** signature(`object = "RTCA timeline"`, `value = "character"`): assign the time unit used by the action track

**start** signature(`object = "RTCA timeline"`): return the starting `POSIXct` time of the experiment

**timeUnit<-** signature(`object = "RTCA timeline"`, `value = "character"`): assign the starting `POSIXct` time of the experiment

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

**See Also**

[RTCA](#)

**Examples**

```
t1 <- new("RTCA timeline")
show(t1)
```

---

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 <a href="#">RTCA</a>
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

```
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 <a href="#">RTCA</a>
...	Parameters passed to <a href="#">smooth.spline</a>

## Details

`smoothTransform` smooths the RTCA cell impedance measurement by calling the function [smooth.spline](#). This feature can be useful for visualisation 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>

## Examples

```
require(RTCA)

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

xSmooth <- smoothTransform(x)
```

---

**spectramaxImport**      *Import output files from Spectramax spectrophotometer*

---

## Description

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

## Usage

```
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**

```
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

\* **IO**  
    parseRTCA, 10

\* **classes**  
    RTCA-class, 16  
    RTCATimeline-class, 18

\* **file**  
    parseRTCA, 10

\* **hplot**  
    controlView, 4  
    plateView, 12  
    plotGridEffect, 13

\* **misc**  
    factor2numeric, 7

\* **models**  
    rgrTransform, 15

\* **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  
    (RTCATimeline-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  
    (RTCATimeline-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

getAction (RTCATimeline-class), 18  
getAction, RTCA, numeric-method  
    (RTCATimeline-class), 16

getAction, RTCATimeline, numeric-method  
    (RTCATimeline-class), 18

grep, 10

interpolationTransform, 7, 8, 15, 16, 20  
interpolationTransform, RTCA-method  
    (RTCATimeline-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, 9, 14, 16, 20  
ratioTransform, RTCA-method  
    (RTCATimeline-class), 16

read.table, 10

relevels, 7  
relevels (factor2numeric), 7

repairAlphaName (alphaNames), 2  
reset (RTCATimeline-class), 18

reset,RTCAtimeline-method  
     (RTCAtimeline-class), 18  
 rgrTransform, 7, 9, 14, 15, 15, 20  
 rgrTransform, RTCA-method (RTCA-class),  
     16  
 rmAction (RTCAtimeline-class), 18  
 rmAction, RTCA, numeric-method  
     (RTCA-class), 16  
 rmAction, RTCAtimeline, numeric-method  
     (RTCAtimeline-class), 18  
 rowcol2pos (alphaNames), 2  
 RTCA, 5, 6, 8–10, 12–15, 18–20  
 RTCA-class, 16  
 RTCAtimeline, RTCA-method (RTCA-class),  
     16  
 RTCAtimeline-class, 18  
 RTCAtimeline<-, RTCA-method  
     (RTCA-class), 16  
  
 show, RTCA-method (RTCA-class), 16  
 sliceRTCA, 19  
 smooth.spline, 20  
 smoothTransform, 7, 9, 15, 16, 20  
 smoothTransform, RTCA, ANY-method  
     (RTCA-class), 16  
 smoothTransform, RTCA, missing-method  
     (RTCA-class), 16  
 spectramaxImport, 21  
 spline, 8  
 startTime (RTCAtimeline-class), 18  
 startTime, RTCAtimeline-method  
     (RTCAtimeline-class), 18  
 startTime<-, (RTCAtimeline-class), 18  
 startTime<-, RTCAtimeline, POSIXct-method  
     (RTCAtimeline-class), 18  
  
 time, RTCA-method (RTCA-class), 16  
 timeline (RTCA-class), 16  
 timeline, RTCA-method (RTCA-class), 16  
 timeline<-, (RTCA-class), 16  
 timelines<-, RTCA-method (RTCA-class), 16  
 timepoints, 10  
 timepoints (RTCA-class), 16  
 timepoints, RTCA-method (RTCA-class), 16  
 timepoints<-, (RTCA-class), 16  
 timepoints<-, RTCA-method (RTCA-class),  
     16  
 timeUnit (RTCAtimeline-class), 18  
  
 timeUnit, RTCAtimeline-method  
     (RTCAtimeline-class), 18  
 timeUnit<-, (RTCAtimeline-class), 18  
 timeUnit<-, RTCAtimeline, character-method  
     (RTCAtimeline-class), 18  
  
 updateAction (RTCAtimeline-class), 18  
 updateAction, RTCA, numeric, character-method  
     (RTCA-class), 16  
 updateAction, RTCAtimeline, numeric, character-method  
     (RTCAtimeline-class), 18