Package 'tbTools'

July 15, 2017

Type Package

Title Tomas' personal mix of utilities

Version 1.0.2

Author Tomas Boril

Maintainer Tomas Boril <borilt@gmail.com>

Description Mix of things that I missed in R. Matlab-
like colon operator, stem plot (base plotting system), round2 with order, ifft etc.

License MIT + file LICENSE

LazyData TRUE

Depends R (>= 3.2.0)

RoxygenNote 6.0.1

Suggests testthat

R topics documented:

ifft *ifft*

Description

Inverse Fast Fourier Transform (discrete FT), Matlab-like behavior.

Usage

```
ifft(sig)
```

Arguments

sig	input vector

Details

This is really the inverse of the fft function, so ifft(fft(x)) == x.

Value

output vector of the same length as the input vector

See Also

fft, Re, Im, Mod, Conj

Examples

```
ifft(fft(1:5))
```

isInt *isInt*

Description

Returns TRUE / FALSE whether it is exactly 1 integer number (in fact, the class can be numeric but the number must be integer), non-missing

Usage

```
isInt(num)
```

Arguments

num	variable to be tested

Value

TRUE / FALSE

See Also

isNum, isLogical, isString

Examples

```
isInt(2)
isInt(2L)
isInt(-2)
isInt(-2L)
isInt(2.1)
isInt(-2.1)
isInt(1:5)
isInt(NA_integer_)
isInt(integer(0))
```

`isLogical` *isLogical*

Description

Returns TRUE / FALSE whether it is exactly 1 logical value, non-missing

Usage

```
isLogical(logical)
```

Arguments

`logical`	variable to be tested

Value

TRUE / FALSE

See Also

isNum, isInt, isString

Examples

```
isLogical(TRUE)
isLogical(FALSE)
isLogical(1)
isLogical(0)
isLogical(2)
isLogical(NA)
isLogical(NaN)
isLogical(logical(0))
```

isNum *isNum*

Description

Returns TRUE / FALSE whether it is exactly 1 number (numeric or integer vector of length 1, non-missing)

Usage

```
isNum(num)
```

Arguments

num	variable to be tested

Value

TRUE / FALSE

See Also

isInt, isLogical, isString

Examples

```
isNum(2)
isNum(2L)
isNum(-2)
isNum(-2L)
isNum(2.1)
isNum(-2.1)
isNum(1:5)
isNum(NA_real_)
isNum(numeric(0))
```

isString *isString*

Description

Returns TRUE / FALSE whether it is exactly 1 character string (character vector of length 1, non-missing)

Usage

```
isString(string)
```

Arguments

string	variable to be tested

Value

TRUE / FALSE

See Also

isInt, isNum, isLogical

Examples

```
isString("hello")
isString(2)
isString(c("hello", "world"))
isString(NA_character_)
```

---

round2 *round2*

Description

Rounds a number to the specified order. Round half away from zero (this is the difference from built-in round function.)

Usage

```
round2(x, order = 0)
```

Arguments

x	number to be rounded
order	0 (default) = units, -1 = 0.1, +1 = 10

Value

rounded number to the specified order

See Also

round, trunc, ceiling, floor

Examples

```
round2(23.5)   # = 24, compare: round(23.5) = 24
round2(23.4)   # = 23
round2(24.5)   # = 25, compare: round(24.5) = 24
round2(-23.5)   # = -24, compare: round(-23.5) = -24
round2(-23.4)   # = -23
round2(-24.5)   # = -25, compare: round(-24.5) = -24
round2(123.456, -1)   # 123.5
round2(123.456, -2)   # 123.46
round2(123.456, 1)  # 120
round2(123.456, 2)  # 100
round2(123.456, 3)  # 0
round2(-123.456, -1)   # -123.5
```

```
round2(-123.456, -2)   # -123.46
round2(-123.456, 1)  # -120
round2(-123.456, 2)  # -100
round2(-123.456, 3)  # 0
```

`seqM` *seqM*

Description

Matlab-like behaviour of colon operator or linspace for creating sequences, for-loop friendly.

Usage

```
seqM(from = NA, to = NA, by = NA, length.out = NA)
```

Arguments

`from`	starting value of the sequence (the first number)
`to`	end value of the sequence (the last number or the boundary number)
`by`	increment of the sequence (if specified, do not use the `length.out` parameter). If both `by` and `length.out` are not specified, then `by = +1`.
`length.out`	desired length of the sequence (if specified, do not use the `by` parameter)

Details

Like `seq()` but with Matlab-like behavior ([: operator] with `by` or [linspace] with `length.out`).

If I create a for-loop, I would like to get an empty vector for 3:1 (I want a default step +1) and also an empty vector for seq(3, 1, by = 1) (not an error). This is solved by this `seqM` function.

Value

returns a vector of type "integer" or "double"

Comparison

R: seqM		Matlab		R: seq
seqM(1, 3)	[1] 1 2 3	1:3	the same	the same
seqM(1, 3, by=.8)	[1] 1.0 1.8 2.6	1:.8:3	the same	the same
seqM(1, 3, by=5)	[1] 1	1:5:3	the same	the same
seqM(3, 1)	integer(0)	3:1	the same	[1] 3 2 1
seqM(3, 1, by=+1)	integer(0)	3:1:1	the same	Error: wrong 'by'
seqM(3, 1, by=-1)	[1] 3 2 1	3:-1:1	the same	the same
seqM(3, 1, by=-3)	[1] 3	3:-3:1	the same	the same
seqM(1, 3, len=5)	[1] 1.0 1.5 2.0 2.5 3.0	linspace(1,3,5)	the same	the same
seqM(1, 3, len=3)	[1] 1 2 3	linspace(1,3,3)	the same	the same
seqM(1, 3, len=2)	[1] 1 3	linspace(1,3,2)	the same	the same
seqM(1, 3, len=1)	[1] 3	linspace(1,3,1)	the same	[1] 1
seqM(1, 3, len=0)	integer(0) + warning	linspace(1,3,0)	the same without warning	the same without warning
seqM(3, 1, len=3)	[1] 3 2 1	linspace(3,1,3)	the same	the same

See Also

round2, isNum, isInt, ifft.

Examples

```
seqM(1, 3)
seqM(1, 3, by=.8)
seqM(1, 3, by=5)
seqM(3, 1)
seqM(3, 1, by=+1)
seqM(3, 1, by=-1)
seqM(3, 1, by=-3)
seqM(1, 3, len=5)
seqM(1, 3, len=3)
seqM(1, 3, len=2)
seqM(1, 3, len=1)
seqM(1, 3, len=0)
seqM(3, 1, len=3)
```

`Stem` *Stem*

Description

Matlab-like stem plotting function for discrete series.

Usage

```
Stem(x, y, pch = 16, linecol = 1, clinecol = 1, ...)
```

Arguments

`x`	horizontal-axis values
`y`	vertical-axis values
`pch`	integer value, style of points (`pch = 21`: circle without fill, see `plot` `pch` parameter)
`linecol`	color of the plot
`clinecol`	zero axis color
`...`	other parameters passed to `plot` function

Details

Discrete plots using base plotting system.

Author: Matti Pastell, Sep 11 2009 http://mpastell.com/2009/09/11/matlab-style-stem-plot-with-r/

Value

creates a plot in base plotting system.

See Also

For interactive time-series plots, see package dygraphs.

Examples

```
t <- seqM(from = 0, to = 2*pi, length.out = 20)
Stem(t, sin(t))
Stem(t, sin(t), pch=21)
Stem(t, sin(t), pch=21, line="blue")
Stem(t, sin(t), main = "Default style")
```

---

strTrim *strTrim*

Description

Trim leading and trailing whitespace in character string.

Usage

```
strTrim(string)
```

Arguments

string	character string

Details

Like str_trim() in stringr package or trimws() in R3.2.0 but way faster.

Source: Hadley Wickham comment at http://stackoverflow.com/questions/2261079/how-to-trim-leading-and-trailing-whitespace-in-r

Value

returns a character string with removed leading and trailing whitespace characters.

See Also

isString for testing whether it is 1 character vector, str_contains for finding string in string without regexp, str_find for all indices without regexp, str_find1 for the first index withoud regexp.

Examples

```
strTrim("      Hello World!    ")
```

str_contains *str_contains*

Description

Find string in another string (without regular expressions), returns TRUE / FALSE.

Usage

```
str_contains(string, patternNoRegex)
```

Arguments

string	string in which we try to find something
patternNoRegex	string we want to find, "as it is" - no regular exprressions

Value

TRUE / FALSE

See Also

str_find, str_find1, isString

Examples

```
str_contains("Hello world", "wor")  # TRUE
str_contains("Hello world", "WOR")  # FALSE
str_contains(tolower("Hello world"), tolower("wor"))  # TRUE
str_contains("Hello world", "")  # TRUE
```

str_find *str_find*

Description

Find string in another string (without regular expressions), returns indices of all occurences.

Usage

```
str_find(string, patternNoRegex)
```

Arguments

string	string in which we try to find something
patternNoRegex	string we want to find, "as it is" - no regular exprressions

Value

indices of all occurences (1 = 1st character)

See Also

str_find1, str_contains, isString

Examples

```
str_find("Hello, hello, hello world", "ell")   # 2 9 16
str_find("Hello, hello, hello world", "q")     # integer(0)
```

---

str_find1 *str_find1*

Description

Find string in another string (without regular expressions), returns indices of the first occurence only.

Usage

```
str_find1(string, patternNoRegex)
```

Arguments

string	string in which we try to find something
patternNoRegex	string we want to find, "as it is" - no regular exprressions

Value

index of the first occurence only (1 = 1st character)

See Also

str_find, str_contains, isString

Examples

```
str_find1("Hello, hello, hello world", "ell")   # 2
str_find1("Hello, hello, hello world", "q")     # integer(0)
```

tbTools *tbTools*

Description

Tomas' personal mix of utilities

Details

Mix of things that I missed in R. Matlab-like colon operator, stem plot (base plotting system), round2 with order, ifft etc.

`seqM` Matlab-like behaviour of colon operator or linspace for creating sequences, for-loop friendly. `round2` Rounds a number to the specified order. Round half away from zero (this is the difference from built-in `round` function.) `ifft` Inverse Fast Fourier Transform (discrete FT), Matlab-like behavior.

`Stem` Matlab-like stem plotting function for discrete series.

`isInt` Returns TRUE / FALSE whether it is exactly 1 integer number (in fact, the class can be numeric but the number must be integer), non-missing `isNum` Returns TRUE / FALSE whether it is exactly 1 number (numeric or integer vector of length 1, non-missing) `isString` Returns TRUE / FALSE whether it is exactly 1 character string (character vector of length 1, non-missing) `isLogical` Returns TRUE / FALSE whether it is exactly 1 logical value (TRUE or FALSE), non-missing

`strTrim` Trim leading and trailing whitespace in character string. Way faster than str_trim() or trimws().

`str_contains` Find string in another string (without regular expressions), returns TRUE / FALSE. `str_find` Find string in another string (without regular expressions), returns indices of all occurences. `str_find1` Find string in another string (without regular expressions), returns indices of the first occurence only.

Index