

R Package `miscset`

User Manual

Sven E. Templer
`sven.templer@gmail.com`

June 6, 2014

Contents

I	Preface	2
1	Introduction	2
2	Installation	2
II	Functions	2
3	Numeric Functions	2
3.1	Generate triangular numbers - <code>ntri</code>	2
3.2	Scale numeric vectors - <code>scale0</code>	2
4	Summarizing	2
4.1	Print a sorted table - <code>sortable</code>	2
5	Data Formatting	3
5.1	Transform to squared matrix - <code>squarematrix</code>	3
5.2	Generate a pairwise list - <code>enpaire</code>	3
6	Data Exporting	4
6.1	Create a latex document containing a table - <code>textable</code>	4
7	Text String Manipulation	4
7.1	Prepend zeroes to unify number lengths - <code>leading0</code>	4
7.2	Extract substrings by pattern - <code>strext</code>	4
7.3	Extract substrings by splitting - <code>strpart</code>	5
7.4	Reverse strings - <code>strrev</code>	5
7.5	Multiple pattern replacement - <code>msub</code> , <code>mgsub</code>	5
8	Pattern Matching	6
8.1	Get index of expression - <code>gregexprind</code>	6
8.2	Multiple pattern search - <code>mgrepl</code>	6
9	Graphical Tools	6
9.1	Create a color palette - <code>gghcl</code>	6
10	System Tools	7
10.1	List details from and remove all objects - <code>lsall</code> , <code>rmall</code>	7

Part I

Preface

1 Introduction

The package **miscset** provides several R tools to read, create, modify and write different types of data. In the following examples, all available functions will be presented including explanations of their usage. Find the source code online at [github](#).

2 Installation

To install the package call the command `install.packages("miscset")` within the R console or run R CMD `install miscset` from a terminal. To use it the `library` or `require` function load the package.

```
require(miscset)

## Loading required package: miscset
```

Part II

Functions

3 Numeric Functions

3.1 Generate triangular numbers - `ntri`

The function generates a series of triangular numbers of length `n` according to [oeis.org](#).

```
ntri(12)

## [1] 0 1 3 6 10 15 21 28 36 45 55 66
```

3.2 Scale numeric vectors - `scale0`

The function scales all values in a numeric vector from 0 to 1.

```
scale0(0:10)

## [1] 0.0 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1.0

scale0(-1:3)

## [1] 0.00 0.25 0.50 0.75 1.00

scale0(2:3)

## [1] 0 1
```

4 Summarizing

4.1 Print a sorted table - `sortable`

Return a sorted table of vectors like `sort(table(x))`. Exclude NA values by choice and sort the table by increasing counts.

```

n <- c(1,2,3,2,2,3,NA)
sortable(n)

## 2 3 1 1
## 3 2 1 1

sortable(n, na = F)

## 2 3 1 1
## 3 2 1 1

sortable(n, decr = F)

## 1 1 3 2
## 1 1 2 3

```

5 Data Formatting

5.1 Transform to squared matrix - squarematrix

The function `squarematrix` can generate a symmetric (square) matrix from an unsymmetric matrix by using the column and row names and filling empty pairs with NA.

```

M <- matrix(1:6, 2, dimnames = list(2:3,1:3))
M

##   1 2 3
## 2 1 3 5
## 3 2 4 6

squarematrix(M)

##   1 2 3
## 1 NA NA NA
## 2 1 3 5
## 3 2 4 6

```

5.2 Generate a pairwise list - enpaire

The function `enpaire` creates a pairwise list of matrix values. The result is a `data.frame` that contains a column for the names of each dimension and the upper and lower triangle values. Unsymmetric matrices are transformed by `squarematrix` (see previous section).

```

M <- matrix(letters[1:9], 3, 3, dimnames = list(1:3,1:3))
M

##   1 2 3
## 1 "a" "d" "g"
## 2 "b" "e" "h"
## 3 "c" "f" "i"

enpaire(M)

##   row col upper lower
## 1 1 2 d b
## 2 1 3 g c
## 3 2 3 h f

```

6 Data Exporting

6.1 Create a latex document containing a table - `textable`

This function enhances the functionality of the `xtable` function from the similar named package. The output of `xtable` is captured, processed and then written to a file. The file contains also latex header for an A4 portrait or landscape article. The function is called with the following syntax: `textable(d, file, caption, rownames, landscape, pt.size, margin, digits, align, label)`

`file` is a character string with the name to the file of the function output. `caption` is a character string with the table's title. `rownames` is logical and allows to switch printing of row names on and off. `landscape` is logical for the orientation of the page, `pt.size` uses an integer value to define the size of the characters. `margin` sets a margin between page and table borders in cm. `digits` defines the number of digits to print for numeric values. With `align` the column alignments can be set. Either a single value or a value for the row names column plus each column in `d`. The possible values are 'r', 'c', 'l' for alignemt to the right, center or left.

7 Text String Manipulation

7.1 Prepend zeroes to unify number lengths - `leading0`

The function `leading0` aims to create e.g. index names with a common string length. It creates character strings from numeric values while attaching 0 in front of the number up to a certain length of total digits of each string.

```
paste0("page", leading0(8:10, 3))

## [1] "page008" "page009" "page010"
```

7.2 Extract substrings by pattern - `strexttr`

The function `strexttr` splits strings in a character vector by `sep` and extracts all substrings matching a given `pattern`.

```
s <- c("a1 b1 c1", "a2 b2", "aa a1", "aa", "b1 a1", "bb ab a1")
strexttr(s, "^[ab][[:digit:]]$")

## [1] NA    NA    "a1"  NA    NA    "a1"

strexttr(s, "^[ab][[:digit:]]$", mult = T)

## [[1]]
## [1] "a1" "b1"
##
## [[2]]
## [1] "a2" "b2"
##
## [[3]]
## [1] "a1"
##
## [[4]]
## [1] NA
##
## [[5]]
## [1] "b1" "a1"
##
## [[6]]
## [1] "a1"
```

```

strexttr(s, "^[ab][[:digit:]]$", mult = T, unlist = T)

## [1] "a1" "b1" "a2" "b2" "a1" NA    "b1" "a1" "a1"

strexttr(s, "^[c][[:digit:]]$")

## [1] "c1" NA    NA    NA    NA    NA

```

7.3 Extract substrings by splitting - strpart

Similar to `strexttr` the function `strpart` supplies a method to extract a substring, but by defining the `nth` part of the string split by the separator given in `sep`.

```

s

## [1] "a1 b1 c1" "a2 b2"    "aa a1"    "aa"        "b1 a1"    "bb ab a1"

strpart(s, " ", 2)

## [1] "b1" "b2" "a1" NA    "a1" "ab"

```

7.4 Reverse strings - strrev

With `strrev` you can create the reversed version of strings.

```

strrev('!dlroW olleH')

## [1] "Hello World!"

s

## [1] "a1 b1 c1" "a2 b2"    "aa a1"    "aa"        "b1 a1"    "bb ab a1"

strrev(s)

## [1] "1c 1b 1a" "2b 2a"    "1a aa"    "aa"        "1a 1b"    "1a ba bb"

```

7.5 Multiple pattern replacement - msub, mgsub

`msub` and `mgsub` behave like `sub` and `gsub` but they replace multiple patterns. Replacement is done in order of the pattern input, and multicore support is enabled by `mclapply` from the parallel package.

```

s

## [1] "a1 b1 c1" "a2 b2"    "aa a1"    "aa"        "b1 a1"    "bb ab a1"

msub("A", "X", s)

## [1] "a1 b1 c1" "a2 b2"    "aa a1"    "aa"        "b1 a1"    "bb ab a1"

mgsub("A", "X", s)

## [1] "a1 b1 c1" "a2 b2"    "aa a1"    "aa"        "b1 a1"    "bb ab a1"

```

8 Pattern Matching

8.1 Get index of expression - gregexprind

```
s
## [1] "a1 b1 c1" "a2 b2"      "aa a1"      "aa"          "b1 a1"      "bb ab a1"

gregexprind("a", s, 1)
## [1] 1 1 1 1 4 4

gregexprind("a", s, 2)
## [1] NA NA  2  2 NA  7

gregexprind("a", s, "last")
## [1] 1 1 4 2 4 7
```

8.2 Multiple pattern search - mgrepl

With `mgrepl(patterns, text, ...)` you can search for more than one regular expression, and use a logical function to combine the results for each single expression.

```
s
## [1] "a1 b1 c1" "a2 b2"      "aa a1"      "aa"          "b1 a1"      "bb ab a1"

mgrepl(c("a", "b"), s, any)
## [1] TRUE TRUE TRUE TRUE TRUE TRUE

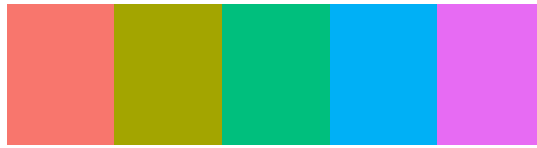
mgrepl(c("a", "b"), s, all)
## [1] TRUE TRUE FALSE FALSE TRUE TRUE
```

9 Graphical Tools

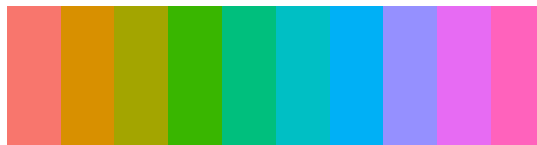
9.1 Create a color palette - gghcl

`gghcl()` creates color palettes. It enhances the `hcl` function. See some examples:

gghcl(5)



gghcl(10)



gghcl(10, 1:5)



10 System Tools

10.1 List details from and remove all objects - `lsall`, `rmall`

With `lsall(envir, ...)` all object names, their length, class, mode and size is returned in a data.frame from a specified environment. `rmall(...)` removes the complete list of objects at the global environment.

```
lsall()
```

```
## Environment: R_GlobalEnv
## Objects:
##   Name Length   Class   Mode  Size Unit
## 1    M      9  matrix character  1.3  Kb
## 2    n      7   numeric   numeric 104.0 byte
## 3    s      6 character character 392.0 byte
```

```
rmall()
```

```
lsall()
```

```
## Environment: R_GlobalEnv
## Objects:
## NULL
```