

# R String Arrays

Fan Wang

2020-08-21

## Contents

<b>1</b>	<b>String Arrays</b>	<b>1</b>
1.1	Positive or Negative Floating Number to String . . . . .	1
1.2	String Replace . . . . .	1
1.3	Search If and Which String Contains . . . . .	2
1.4	String Split . . . . .	3
1.5	String Concatenate . . . . .	3
1.6	String Add Leading Zero . . . . .	3
1.7	Substring Components . . . . .	3

## 1 String Arrays

Go to the [RMD](#), [R](#), [PDF](#), or [HTML](#) version of this file. Go back to [fan's REconTools Package](#), [R Code Examples Repository \(bookdown site\)](#), or [Intro Stats with R Repository \(bookdown site\)](#).

### 1.1 Positive or Negative Floating Number to String

There is a number, that contains decimal and possibly negative sign and has some decimals, convert this to a string that is more easily used as a file or folder name.

```
ls_fl_rho <- c(1, -1, -1.5 -100, 0.5, 0.11111111, -199.22123)
for (fl_rho in ls_fl_rho) {
  st_rho <- paste0(round(fl_rho, 4))
  st_rho <- gsub(x = st_rho, pattern = "-", replacement = "n")
  st_rho <- gsub(x = st_rho, pattern = "\\.", replacement = "p")
  print(paste0('st_rho=', st_rho))
}
```

```
## [1] "st_rho=1"
## [1] "st_rho=n1"
## [1] "st_rho=n101p5"
## [1] "st_rho=0p5"
## [1] "st_rho=0p1111"
## [1] "st_rho=n199p2212"
```

### 1.2 String Replace

- r string wildcard replace between regex
- [R - replace part of a string using wildcards](#)

```
# String replacement
gsub(x = paste0(unique(df.slds.stats.perc$it.inner.counter), ':'),
```

```

      unique(df.slds.stats.perc$z_n_a_n), collapse = ';'),
    pattern = "\n",
    replacement = "")
gsub(x = var, pattern = "\n", replacement = "")
gsub(x = var.input, pattern = "\\.", replacement = "_")

```

String replaces a segment, search by wildcard. Given the string below, delete all text between carriage return and pound sign:

```

st_tex_text <- "\n% Lat2ex Comments\n\\newcommand{\exa}{\text{from external file: } \\alpha + \\beta}
st_clean_a1 <- gsub("\\%.*?\\n", "", st_tex_text)
st_clean_a2 <- gsub("L.*?x", "[LATEX]", st_tex_text)
print(paste0('st_tex_text:', st_tex_text))

```

```

## [1] "st_tex_text:\n% Lat2ex Comments\n\\newcommand{\exa}{\text{from external file: } \\alpha + \\beta}
print(paste0('st_clean_a1:', st_clean_a1))

```

```

## [1] "st_clean_a1:\n\\newcommand{\exa}{\text{from external file: } \\alpha + \\beta}\n"
print(paste0('st_clean_a2:', st_clean_a2))

```

```

## [1] "st_clean_a2:\n% [LATEX] Comments\n\\newcommand{\exa}{\text{from external file: } \\alpha + \\beta}\n"

```

String delete after a particular string:

```

st_tex_text <- "\\end{equation}\n\n% Even more comments from Latex preamble"
st_clean_a1 <- gsub("\\n%.*", "", st_tex_text)
print(paste0('st_tex_text:', st_tex_text))

```

```

## [1] "st_tex_text:\\end{equation}\n\n% Even more comments from Latex preamble"
print(paste0('st_clean_a1:', st_clean_a1))

```

```

## [1] "st_clean_a1:\\end{equation}\n"

```

### 1.3 Search If and Which String Contains

- r if string contains
- r if string contains either or grepl
- [Use grepl to search either of multiple substrings in a text](#)

Search for a single substring in a single string:

```

st_example_a <- 'C:/Users/fan/R4Econ/amto/tibble/fs_tib_basics.Rmd'
st_example_b <- 'C:/Users/fan/R4Econ/amto/tibble/_main.html'
grepl('_main', st_example_a)

```

```

## [1] FALSE

```

```

grepl('_main', st_example_b)

```

```

## [1] TRUE

```

Search for if one of a set of substring exists in a set of strings. In particular which one of the elements of *ls\_spn* contains at least one of the elements of *ls\_str\_if\_contains*. In the example below, only the first path does not contain either the word *aggregate* or *index* in the path. This can be used after all paths have been found recursively in some folder to select only desired paths from the full set of possibilities:

```

ls_spn <- c("C:/Users/fan/R4Econ//panel/basic/fs_genpanel.Rmd",
            "C:/Users/fan/R4Econ//summarize/aggregate/main.Rmd",

```

```

"C:/Users/fan/R4Econ//summarize/index/fs_index_populate.Rmd")
ls_str_if_contains <- c("aggregate", "index")
str_if_contains <- paste(ls_str_if_contains, collapse = "|")
grepl(str_if_contains, ls_spn)

```

```
## [1] FALSE TRUE TRUE
```

## 1.4 String Split

Given some string, generated for example by cut, get the lower cut starting points, and also the higher end point

```

# Extract 0.216 and 0.500 as lower and upper bounds
st_cut_cate <- '(0.216,0.500]'
# Extract Lower Part
substring(strsplit(st_cut_cate, ",")[[1]][1], 2)

```

```
## [1] "0.216"
```

```

# Extract second part except final bracket Option 1
intToUtf8(rev(utf8ToInt(substring(intToUtf8(rev(utf8ToInt(strsplit(st_cut_cate, ",")[[1]][2])), 2))))

```

```
## [1] "0.500"
```

```

# Extract second part except final bracket Option 2
gsub(strsplit(st_cut_cate, ",")[[1]][2], pattern = "]", replacement = "")

```

```
## [1] "0.500"
```

## 1.5 String Concatenate

```

# Simple Collapse
vars.group.by <- c('abc', 'efg')
paste0(vars.group.by, collapse='|')

```

```
## [1] "abc|efg"
```

## 1.6 String Add Leading Zero

```

# Add Leading zero for integer values to allow for sorting when
# integers are combined into strings
it_z_n <- 1
it_a_n <- 192
print(sprintf("%02d", it_z_n))

```

```
## [1] "01"
```

```
print(sprintf("%04d", it_a_n))
```

```
## [1] "0192"
```

## 1.7 Substring Components

Given a string, with certain structure, get components.

- r time string get month and year and day

```
snm_full <- "20100701"
snm_year <-substr(snm_full,0,4)
snm_month <-substr(snm_full,5,6)
snm_day <-substr(snm_full,7,8)
print(paste0('full:', snm_full,
             ', year:', snm_year,
             ', month:', snm_month,
             ', day:', snm_day))

## [1] "full:20100701, year:2010, month:07, day:01"
```