# Apply the Same Function over Columns and Row Groups

Fan Wang

2023-03-20

## Contents



# 1 Apply Function Over Multiple Columns and Rows

Go to the **RMD**, **R**, **PDF**, or **HTML** version of this file. Go back to fan's REconTools research support package, R4Econ examples page, PkgTestR packaging guide, or Stat4Econ course page.

## 1.1 Convert Subset of Variables to Numeric

Multiply a subset of variables all by 10. We use dplyr's across function to achieve this.

Note that in the example below, we also use across with group by to include a string array of grouping by variables.

Note: "across() makes it easy to apply the same transformation to multiple columns, allowing you to use select() semantics inside in"data-masking" functions like summarise() and mutate()."

```
# grouping by variables
ar_st_groups <- c("gear", "carb")
# use across to conduct operation over multiple variables
mtcars_3var_times10 <- mtcars %>%
  group_by(across(one_of(ar_st_groups))) %>%
  mutate(across(matches("mpg|cyl|disp"), ~ .x * 10)) %>%
  select(gear, carb, mpg, cyl, disp) %>% head(n=5)
# pring
# Multiply several variables by 10
kable(mtcars_3var_times10 %>% slice_head(n = 5)) %>%
    kable_styling_fc()
```

## 1.2 Compute Row-specific Quantiles using Data Across Columns

We want to compute quantiles for each location, based on monthly variations in columns.

First, we generate a table with 12 columns (for months) and 3 rows (for locations).

| gear | carb | mpg | cyl | disp |
|---|---|---|---|---|
| 3 | 1 | 214 | 60 | 2580 |
| 3 | 2 | 187 | 80 | 3600 |
| 4 | 1 | 228 | 40 | 1080 |
| 4 | 4 | 210 | 60 | 1600 |
| 4 | 4 | 210 | 60 | 1600 |

```
# Generate data, 12 months as columns, and
mt_data_rand <- matrix(rnorm(36, mean=0, sd=1), nrow=3, ncol=12)
it_rows <- seq(1, dim(mt_data_rand)[1])
it_cols <- seq(1, dim(mt_data_rand)[2])
# convert to table, column as month with leading 0
colnames(mt_data_rand) <- paste0('m', sprintf("%02d", it_cols))
tb_data_full <- as_tibble(mt_data_rand, rownames = NA) %>%
    mutate(loc = paste0("loc", sprintf("%02d", row_number()))) %>%
    select(loc, everything())
# Display
kable(tb_data_full) %>% kable_styling_fc_wide()
```

| loc | m01 | m02 | m03 | m04 | m05 | m06 | m07 | m08 | m09 | m10 | m11 | m12 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| loc01 | 0.4007715 | 1.7869131 | 0.7013559 | -0.2179749 | -0.6250393 | 0.1533731 | 0.4264642 | 0.8781335 | 0.5539177 | -0.3804710 | -1.265396 | -1.1231086 |
| loc02 | 0.1106827 | 0.4978505 | -0.4727914 | -1.0260044 | -1.6866933 | -1.1381369 | -0.2950715 | 0.8215811 | -0.0619117 | -0.6947070 | 2.168956 | -0.4028848 |
| loc03 | -0.5558411 | -1.9666172 | -1.0678237 | -0.7288912 | 0.8377870 | 1.2538149 | 0.8951257 | 0.6886403 | -0.3059627 | -0.2079173 | 1.207962 | -0.4666554 |

Second, using apply to compute quantiles, row by row

```
# Extract the data components from the tibble, tibble has row and column names
tb_data_only <- tb_data_full %>%
    column_to_rownames(var = "loc") %>%
    select(contains("m"))
# Compute row specific quantiles
ar_quantiles_by_row <- apply(tb_data_only, 1, quantile, probs=0.75)
# Display
print(ar_quantiles_by_row)
```

```
##     loc01     loc02     loc03
## 0.5907772 0.2074747 0.8521217
```

Third, generate matrix of two columns, ID and quantile.

```
# One particular quantil from location
tb_loc_quantile <- as_tibble(ar_quantiles_by_row) %>%
    mutate(loc = names(ar_quantiles_by_row)) %>%
    rename(quantile = value) %>%
    select(loc, everything())
# Display
kable(tb_loc_quantile) %>% kable_styling_fc()
```

| loc | quantile |
|---|---|
| loc01 | 0.5907772 |
| loc02 | 0.2074747 |
| loc03 | 0.8521217 |

## 1.3 Compute Row-specific Sums using Data Across Columns

We compute sum over several variables in the mtcars dataset. We will sum over several variables with shared prefix, after adding these prefix first. We introduce an NA value to make sure that we can sum ignoring NA

We sum using three different methods below: (1) purrr:reduce(), (2) base::rowSums(), (3) Manual sum. Note that the rowSums option is able to sum ignoring NA.

```
# we introduce NA value to first row
mtcars[1,1] <- NA
# Rename variables, and sum across
mtcars_rowsum <- mtcars %>%
    rename(stats_mpg = mpg, stats_cyl = cyl, stats_hp = hp) %>%
    mutate(
        cs_reduce = purrr::reduce(
            dplyr::pick(contains("stats")),
            `+`
        ),
        cs_rowsum = base::rowSums(
            dplyr::pick(contains("stats")),
            na.rm = TRUE
        ),
        cs_manual = stats_mpg + stats_cyl + stats_hp
    ) %>%
    select(matches("stats|cs"), gear)
# Display
# caption: "sum across columns"
kable(mtcars_rowsum %>% slice_head(n = 5)) %>% kable_styling_fc_wide()
```

| | stats_mpg | stats_cyl | stats_hp | cs_reduce | cs_rowsum | cs_manual | gear |
|---|---|---|---|---|---|---|---|
| Mazda RX4 | NA | 6 | 110 | NA | 116.0 | NA | 4 |
| Mazda RX4 Wag | 21.0 | 6 | 110 | 137.0 | 137.0 | 137.0 | 4 |
| Datsun 710 | 22.8 | 4 | 93 | 119.8 | 119.8 | 119.8 | 4 |
| Hornet 4 Drive | 21.4 | 6 | 110 | 137.4 | 137.4 | 137.4 | 3 |
| Hornet Sportabout | 18.7 | 8 | 175 | 201.7 | 201.7 | 201.7 | 3 |

See this discussion for column sum peed comparisons.

## 1.4 Sum Across Rows within Group

Following from the prior section, we now sum across rows within group.

```
# we introduce NA value to first row
# mtcars[1,1] <- NA
# Rename variables, and sum across
mtcars_grpsum <- mtcars_rowsum %>%
    arrange(gear) %>% group_by(gear) %>%
    # srs = sum row sum
    mutate_at(vars(matches("stats|cs")),
        .funs = list(gs = ~sum(., na.rm=TRUE))
    ) %>%
    select(gear, matches("gs")) %>%
    slice_head(n=1)
# Display
# caption: "gs = group sum, cs = col sum over the columns
# with stats as prefix, sum across rows after col sum; gear = 4
```

```
# difference for cs-rowsum-gs because it allowed for summing
# ignoring NA for values across columns"
kable(mtcars_grpsum) %>% kable_styling_fc_wide()
```

| gear | stats__mpg__gs | stats__cyl__gs | stats__hp__gs | cs__reduce__gs | cs__rowsum__gs | cs__manual__gs |
|---|---|---|---|---|---|---|
| 3 | 241.6 | 112 | 2642 | 2995.6 | 2995.6 | 2995.6 |
| 4 | 273.4 | 56 | 1074 | 1287.4 | 1403.4 | 1287.4 |
| 5 | 106.9 | 30 | 978 | 1114.9 | 1114.9 | 1114.9 |

## 1.5 Replace NA for Multiple Variables

Replace some variables NA by some values, and other variables' NAs by other values.

```
# Define
it_N <- 3
it_M <- 5
svr_id <- "date"

# NA dataframe, note need to define as NA_real_
# if define as NA, will not be able to replace with 99 column
# would be logical rather than double.
df_NA <- as_tibble(matrix(NA_real_, nrow = it_N, ncol = it_M)) %>%
    rowid_to_column(var = svr_id) %>%
    rename_at(
        vars(starts_with("V")),
        funs(str_replace(., "V", "var"))
    )
kable(df_NA) %>%
    kable_styling_fc()
```

```
# Replace NA
df_NA_replace <- df_NA %>%
    mutate_at(vars(one_of(c("var1", "var2"))), list(~ replace_na(., 0))) %>%
    mutate_at(vars(one_of(c("var3", "var5"))), list(~ replace_na(., 99)))

kable(df_NA_replace) %>%
    kable_styling_fc()
```

## 1.6 Cumulative Sum Multiple Variables

Each row is a different date, each column is the profit a firms earns on a date, we want to compute cumulatively how much a person is earning. Also renames variable names below jointly.

```
# Define
it_N <- 3
it_M <- 5
svr_id <- "date"
```

| date | var1 | var2 | var3 | var4 | var5 |
|---|---|---|---|---|---|
| 1 | NA | NA | NA | NA | NA |
| 2 | NA | NA | NA | NA | NA |
| 3 | NA | NA | NA | NA | NA |

| date | var1 | var2 | var3 | var4 | var5 |
|---|---|---|---|---|---|
| 1 | 0 | 0 | 99 | NA | 99 |
| 2 | 0 | 0 | 99 | NA | 99 |
| 3 | 0 | 0 | 99 | NA | 99 |

| date | dp_f1 | dp_f2 | dp_f3 | dp_f4 | dp_f5 |
|---|---|---|---|---|---|
| 1 | -0.5604756 | 0.0705084 | 0.4609162 | -0.4456620 | 0.4007715 |
| 2 | -0.2301775 | 0.1292877 | -1.2650612 | 1.2240818 | 0.1106827 |
| 3 | 1.5587083 | 1.7150650 | -0.6868529 | 0.3598138 | -0.5558411 |

```
# random dataframe, daily profit of firms
# dp_fx: daily profit firm ID something
set.seed(123)
df_daily_profit <- as_tibble(matrix(rnorm(it_N * it_M), nrow = it_N, ncol = it_M)) %>%
    rowid_to_column(var = svr_id) %>%
    rename_at(
        vars(starts_with("V")),
        funs(str_replace(., "V", "dp_f"))
    )
kable(df_daily_profit) %>%
    kable_styling_fc()
```

```
# cumulative sum with suffix
df_cumu_profit_suffix <- df_daily_profit %>%
    mutate_at(vars(contains("dp_f")), .funs = list(cumu = ~ cumsum(.)))
kable(df_cumu_profit_suffix) %>%
    kable_styling_fc_wide()
```

| date | dp_f1 | dp_f2 | dp_f3 | dp_f4 | dp_f5 | dp_f1_cumu | dp_f2_cumu | dp_f3_cumu | dp_f4_cumu | dp_f5_cumu |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | -0.5604756 | 0.0705084 | 0.4609162 | -0.4456620 | 0.4007715 | -0.5604756 | 0.0705084 | 0.4609162 | -0.4456620 | 0.4007715 |
| 2 | -0.2301775 | 0.1292877 | -1.2650612 | 1.2240818 | 0.1106827 | -0.7906531 | 0.1997961 | -0.8041450 | 0.7784198 | 0.5114542 |
| 3 | 1.5587083 | 1.7150650 | -0.6868529 | 0.3598138 | -0.5558411 | 0.7680552 | 1.9148611 | -1.4909979 | 1.1382337 | -0.0443870 |

```
# cumulative sum variables naming to prefix
df_cumu_profit <- df_cumu_profit_suffix %>%
    rename_at(vars(contains("_cumu")), list(~ paste("cp_f", gsub("_cumu", "", .), sep = ""))) %>%
    rename_at(vars(contains("cp_f")), list(~ gsub("dp_f", "", .)))
kable(df_cumu_profit) %>%
    kable_styling_fc_wide()
```

| date | dp_f1 | dp_f2 | dp_f3 | dp_f4 | dp_f5 | cp_f1 | cp_f2 | cp_f3 | cp_f4 | cp_f5 |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | -0.5604756 | 0.0705084 | 0.4609162 | -0.4456620 | 0.4007715 | -0.5604756 | 0.0705084 | 0.4609162 | -0.4456620 | 0.4007715 |
| 2 | -0.2301775 | 0.1292877 | -1.2650612 | 1.2240818 | 0.1106827 | -0.7906531 | 0.1997961 | -0.8041450 | 0.7784198 | 0.5114542 |
| 3 | 1.5587083 | 1.7150650 | -0.6868529 | 0.3598138 | -0.5558411 | 0.7680552 | 1.9148611 | -1.4909979 | 1.1382337 | -0.0443870 |