R Generate Arrays

Fan Wang

2021-03-23

Contents

1 Generate 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 Generate Often Used Arrays

1.1.1 Equi-distance Array with Bound

Consider multiple income groups in income bins that are equal-width, for the final income group, consider all individuals above some final bin minimum bound. Below the code generates this array of numbers: $0, 20000, 40000, 60000, 80000, 100000, 100000000$.

```
# generate income cut-offs
fl_bin_start <- 0
# width equal to 20,000
fl_bin_width <- 2e4
# final point is 100 million
fl_bin_final_end <- 1e8
# final segment starting point is 100,000 dollars
fl_bin_final_start <- 1e5
# generate tincome bins
ar_income_bins <- c(seq(fl_bin_start, fl_bin_final_start, by=fl_bin_width),
                    fl_bin_final_end)
# Display
print(ar_income_bins)
```

```
## [1] 0e+00 2e+04 4e+04 6e+04 8e+04 1e+05 1e+08
```

Generate finer bins, at 5000 USD intervals, and stopping at 200 thousand dollars.

```
fl_bin_start <- 0
fl_bin_width <- 5e3
fl_bin_final_end <- 1e8
fl_bin_final_start <- 2e5
ar_income_bins <- c(seq(fl_bin_start, fl_bin_final_start, by=fl_bin_width),
```

```
                    fl_bin_final_end)
print(ar_income_bins)
```

```
##  [1] 0.00e+00 5.00e+03 1.00e+04 1.50e+04 2.00e+04 2.50e+04 3.00e+04 3.50e+04 4.00e+04 4.50e+04 5.00e
## [15] 7.00e+04 7.50e+04 8.00e+04 8.50e+04 9.00e+04 9.50e+04 1.00e+05 1.05e+05 1.10e+05 1.15e+05 1.20e
## [29] 1.40e+05 1.45e+05 1.50e+05 1.55e+05 1.60e+05 1.65e+05 1.70e+05 1.75e+05 1.80e+05 1.85e+05 1.90e
```

1.1.2 Log Space Arrays

Often need to generate arrays on log rather than linear scale, below is log 10 scaled grid.

```
# Parameters
it.lower.bd.inc.cnt <- 3
fl.log.lower <- -10
fl.log.higher <- -9
fl.min.rescale <- 0.01
it.log.count <- 4
# Generate
ar.fl.log.rescaled <- exp(log(10)*seq(log10(fl.min.rescale),
                                      log10(fl.min.rescale +
                                              (fl.log.higher-fl.log.lower)),
                                      length.out=it.log.count))
ar.fl.log <- ar.fl.log.rescaled + fl.log.lower - fl.min.rescale
# Print
ar.fl.log
```

```
## [1] -10.000000  -9.963430  -9.793123  -9.000000
```