Examples of Lists and Named One and Two Dimensional Lists in R

Fan Wang

2020-04-13

Contents

Multiple Dimensional List

Go to the **RMD**, **R**, **PDF**, or **HTML** version of this file. Go back to fan's REconTools Package, R4Econ Repository (bookdown site), or Intro Stats with R Repository.

- r list tutorial
- r vector vs list
- r initialize empty multiple element list
- r name rows and columns of 2 dimensional list
- r row and colum names of list
- list dimnames

One Dimensional Named List

1. define list
2. slice list

```
# Define Lists
ls_num <- list(1,2,3)
ls_str <- list('1','2','3')
ls_num_str <- list(1,2,'3')

# Named Lists
ar_st_names <- c('e1','e2','e3')
ls_num_str_named <- ls_num_str
names(ls_num_str_named) <- ar_st_names

# Add Element to Named List
ls_num_str_named$e4 <- 'this is added'

# display
print(paste0('ls_num:', str(ls_num)))
```

```
## List of 3
##  $ : num 1
##  $ : num 2
##  $ : num 3
## [1] "ls_num:"
```

```
print(paste0('ls_num[2:3]:', str(ls_num[2:3])))
```

```
## List of 2
##  $ : num 2
##  $ : num 3
## [1] "ls_num[2:3]:"
```

```
print(paste0('ls_str:', str(ls_str)))
```

```
## List of 3
##  $ : chr "1"
##  $ : chr "2"
##  $ : chr "3"
## [1] "ls_str:"
```

```
print(paste0('ls_str[2:3]:', str(ls_str[2:3])))
```

```
## List of 2
##  $ : chr "2"
##  $ : chr "3"
## [1] "ls_str[2:3]:"
```

```
print(paste0('ls_num_str:', str(ls_num_str)))
```

```
## List of 3
##  $ : num 1
##  $ : num 2
##  $ : chr "3"
## [1] "ls_num_str:"
```

```
print(paste0('ls_num_str[2:4]:', str(ls_num_str[2:4])))
```

```
## List of 3
##  $ : num 2
##  $ : chr "3"
##  $ : NULL
## [1] "ls_num_str[2:4]:"
```

```
print(paste0('ls_num_str_named:', str(ls_num_str_named)))
```

```
## List of 4
##  $ e1: num 1
##  $ e2: num 2
##  $ e3: chr "3"
##  $ e4: chr "this is added"
## [1] "ls_num_str_named:"
```

```
print(paste0('ls_num_str_named[c(\'e2\',\'e3\',\'e4\')]', str(ls_num_str_named[c('e2','e3','e4')])))
```

```
## List of 3
##  $ e2: num 2
##  $ e3: chr "3"
##  $ e4: chr "this is added"
## [1] "ls_num_str_named[c('e2','e3','e4')]"
```

Two Dimensional Unnamed List Generate a multiple dimensional list:

1. Initiate with an N element empty list
2. Reshape list to M by Q
3. Fill list elements
4. Get list element by row and column number

List allows for different data types to be stored together.

Note that element specific names in named list are not preserved when the list is reshaped to be two dimensional. Two dimensional list, however, could have row and column names.

```
# Dimensions
it_M <- 2
it_Q <- 3
it_N <- it_M*it_Q

# Initiate an Empty MxQ=N element list
ls_2d_flat <- vector(mode = "list", length = it_N)
ls_2d <- ls_2d_flat

# Named flat
ls_2d_flat_named <- ls_2d_flat
names(ls_2d_flat_named) <- paste0('e',seq(1,it_N))
ls_2d_named <- ls_2d_flat_named

# Reshape
dim(ls_2d) <- c(it_M, it_Q)
# named 2d list can not carry 1d name after reshape
dim(ls_2d_named) <- c(it_M, it_Q)
```

Print Various objects generated above:

```
# display
print('ls_2d_flat')
```

```
## [1] "ls_2d_flat"
```

```
print(ls_2d_flat)
```

```
## [[1]]
## NULL
##
## [[2]]
## NULL
##
## [[3]]
## NULL
##
## [[4]]
## NULL
##
## [[5]]
## NULL
##
## [[6]]
## NULL
```

```
print('ls_2d_flat_named')
```

```
## [1] "ls_2d_flat_named"
```

```
print(ls_2d_flat_named)
```

```
## $e1
```

```
## NULL
##
## $e2
## NULL
##
## $e3
## NULL
##
## $e4
## NULL
##
## $e5
## NULL
##
## $e6
## NULL
```

```
print('ls_2d')
```

```
## [1] "ls_2d"
```

```
print(ls_2d)
```

```
##      [,1] [,2] [,3]
## [1,] NULL NULL NULL
## [2,] NULL NULL NULL
```

```
print('ls_2d_named')
```

```
## [1] "ls_2d_named"
```

```
print(ls_2d_named)
```

```
##      [,1] [,2] [,3]
## [1,] NULL NULL NULL
## [2,] NULL NULL NULL
```

Select element from list:

```
# Select Values, double bracket to select from 2dim list
print('ls_2d[[1,2]]')
```

```
## [1] "ls_2d[[1,2]]"
```

```
print(ls_2d[[1,2]])
```

```
## NULL
```

Define Two Dimensional Named LIst For naming two dimensional lists, *rowname* and *colname* does not work. Rather, we need to use *dimnames*. Note that in addition to dimnames, we can continue to have element specific names. Both can co-exist. But note that the element specific names are not preserved after dimension transform, so need to be redefined afterwards.

How to select an element of a two dimensional list:

1. row and column names: *dimnames*, *ls_2d_flat_named[['row2','col2']]*
2. named elements: *names*, *ls_2d_flat_named[['e5']]*
3. select by index: *index*, *ls_2d_flat_named[[5]]*
4. converted two dimensional named list to tibble/matrix

Neither *dimnames* nor *names* are required, but both can be used to select elements.

```
# Dimensions
it_M <- 3
it_Q <- 4
it_N <- it_M*it_Q

# Initiate an Empty MxQ=N element list
ls_2d_flat_named <- vector(mode = "list", length = it_N)
dim(ls_2d_flat_named) <- c(it_M, it_Q)

# Fill with values
for (it_Q_ctr in seq(1,it_Q)) {
  for (it_M_ctr in seq(1,it_M)) {
    # linear index
    ls_2d_flat_named[[it_M_ctr, it_Q_ctr]] <- (it_Q_ctr-1)*it_M+it_M_ctr
  }
}

# Replace row names, note rownames does not work
dimnames(ls_2d_flat_named)[[1]] <- paste0('row',seq(1,it_M))
dimnames(ls_2d_flat_named)[[2]] <- paste0('col',seq(1,it_Q))

# Element Specific Names
names(ls_2d_flat_named) <- paste0('e',seq(1,it_N))

# Convert to Matrix
tb_2d_flat_named <- as_tibble(ls_2d_flat_named) %>% unnest()
mt_2d_flat_named <- as.matrix(tb_2d_flat_named)
```

Print various objects generated above:

```
# These are not element names, can still name each element
# display
print('ls_2d_flat_named')
```

```
## [1] "ls_2d_flat_named"
```

```
print(ls_2d_flat_named)
```

```
##      col1 col2 col3 col4
## row1 1    4    7    10
## row2 2    5    8    11
## row3 3    6    9    12
## attr(,"names")
##  [1] "e1"  "e2"  "e3"  "e4"  "e5"  "e6"  "e7"  "e8"  "e9"  "e10" "e11" "e12"
```

```
print('str(ls_2d_flat_named)')
```

```
## [1] "str(ls_2d_flat_named)"
```

```
print(str(ls_2d_flat_named))
```

```
## List of 12
##  $ e1 : num 1
##  $ e2 : num 2
##  $ e3 : num 3
##  $ e4 : num 4
```

```
##  $ e5 : num 5
##  $ e6 : num 6
##  $ e7 : num 7
##  $ e8 : num 8
##  $ e9 : num 9
##  $ e10: num 10
##  $ e11: num 11
##  $ e12: num 12
##  - attr(*, "dim")= int [1:2] 3 4
##  - attr(*, "dimnames")=List of 2
##   ..$ : chr [1:3] "row1" "row2" "row3"
##   ..$ : chr [1:4] "col1" "col2" "col3" "col4"
## NULL
```

```
print('tb_2d_flat_named')
```

```
## [1] "tb_2d_flat_named"
```

```
print(tb_2d_flat_named)
```

```
## # A tibble: 3 x 4
##    col1  col2  col3  col4
##   <dbl> <dbl> <dbl> <dbl>
## 1     1     4     7    10
## 2     2     5     8    11
## 3     3     6     9    12
```

```
print('mt_2d_flat_named')
```

```
## [1] "mt_2d_flat_named"
```

```
print(mt_2d_flat_named)
```

```
##      col1 col2 col3 col4
## [1,]    1    4    7   10
## [2,]    2    5    8   11
## [3,]    3    6    9   12
```

Select elements from list:

```
# Select elements with with dimnames
print('ls_2d_flat_named[[\'row2\',\'col2\']]')
```

```
## [1] "ls_2d_flat_named[['row2','col2']]"
```

```
print(ls_2d_flat_named[['row2','col2']])
```

```
## [1] 5
```

```
# Select elements with element names
print('ls_2d_flat_named[[\'e5\']]')
```

```
## [1] "ls_2d_flat_named[['e5']]"
```

```
print(ls_2d_flat_named[['e5']])
```

```
## [1] 5
```

```
# Select elements with index
print('ls_2d_flat_named[[5]]')
```

```
## [1] "ls_2d_flat_named[[5]]"
```

```
print(ls_2d_flat_named[[5]])
```

```
## [1] 5
```