R Generate and Combine Fixed and Random Matrix

Fan Wang

2022-07-14

Contents

1 Generate Matrixes 1
1.1 Create a N by 2 Matrix from 3 arrays . . . 1
1.2 Name Matrix Columns and Rows . . . 1
1.3 Generate NA Matrix . . . 1
1.4 Generate Random Matrixes . . . 2
1.5 Sort Each Matrix Row or Column . . . 2
1.6 Compute Column and Row Statistics . . . 3
1.7 Add Column to Matrix with Common Scalar Value . . . 4

1 Generate Matrixes

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 Create a N by 2 Matrix from 3 arrays

Names of each array become row names automatically.

```
ar_row_one <- c(-1,+1)
ar_row_two <- c(-3,-2)
ar_row_three <- c(0.35,0.75)

mt_n_by_2 <- rbind(ar_row_one, ar_row_two, ar_row_three)
kable(mt_n_by_2) %>%
  kable_styling_fc()
```

ar_row_one	-1.00	1.00
ar_row_two	-3.00	-2.00
ar_row_three	0.35	0.75

1.2 Name Matrix Columns and Rows

```
# An empty matrix with Logical NA
mt_named <- matrix(data=NA, nrow=2, ncol=2)
colnames(mt_named) <- paste0('c', seq(1,2))
rownames(mt_named) <- paste0('r', seq(1,2))
mt_named
```

```
##    c1 c2
## r1 NA NA
## r2 NA NA
```

1.3 Generate NA Matrix

- Best way to allocate matrix in R, NULL vs NA?

Allocate with NA or NA_real_ or NA_int_. Clarity in type definition is preferred.

```
# An empty matrix with Logical NA
mt_na <- matrix(data=NA, nrow=2, ncol=2)
str(mt_na)
```

```
##  logi [1:2, 1:2] NA NA NA NA
```

```
# An empty matrix with numerica NA
mt_fl_na <- matrix(data=NA_real_, nrow=2, ncol=2)
mt_it_na <- matrix(data=NA_integer_, nrow=2, ncol=2)

str(mt_fl_na)
```

```
##  num [1:2, 1:2] NA NA NA NA
```

```
str(mt_fl_na)
```

```
##  num [1:2, 1:2] NA NA NA NA
```

1.4 Generate Random Matrixes

Random draw from the normal distribution, random draw from the uniform distribution, and combine resulting matrixes.

```
# Generate 15 random normal, put in 5 rows, and 3 columns
mt_rnorm <- matrix(rnorm(15,mean=0,sd=1), nrow=5, ncol=3)

# Generate 15 random normal, put in 5 rows, and 3 columns
mt_runif <- matrix(runif(15,min=0,max=1), nrow=5, ncol=5)

# Combine
mt_rnorm_runif <- cbind(mt_rnorm, mt_runif)

# Display
kable(round(mt_rnorm_runif, 3)) %>% kable_styling_fc()
```

-1.690	1.281	0.549	0.478	0.143	0.139	0.478	0.143
1.239	-1.727	0.238	0.758	0.415	0.233	0.758	0.415
-0.109	1.690	-1.049	0.216	0.414	0.466	0.216	0.414
-0.117	0.504	1.295	0.318	0.369	0.266	0.318	0.369
0.183	2.528	0.826	0.232	0.152	0.858	0.232	0.152

1.5 Sort Each Matrix Row or Column

Now we sort within each row or within each column of the random matrix.

```
# Within row sort
mt_rnorm_runif_row_sort <- t(apply(
  mt_rnorm_runif, 1, sort
))
```

```r
# Within column sort, note no transpose
mt_rnorm_runif_col_sort <- apply(
  mt_rnorm_runif, 2, sort
)
# Display
kable(round(mt_rnorm_runif_row_sort, 3),
      caption="Each row sort low to high") %>%
  kable_styling_fc()
```

Each row sort low to high

-1.690	0.139	0.143	0.143	0.478	0.478	0.549	1.281
-1.727	0.233	0.238	0.415	0.415	0.758	0.758	1.239
-1.049	-0.109	0.216	0.216	0.414	0.414	0.466	1.690
-0.117	0.266	0.318	0.318	0.369	0.369	0.504	1.295
0.152	0.152	0.183	0.232	0.232	0.826	0.858	2.528

```r
kable(round(mt_rnorm_runif_col_sort, 3),
      caption="Each column sort low to high") %>%
  kable_styling_fc()
```

Each column sort low to high

-1.690	-1.727	-1.049	0.216	0.143	0.139	0.216	0.143
-0.117	0.504	0.238	0.232	0.152	0.233	0.232	0.152
-0.109	1.281	0.549	0.318	0.369	0.266	0.318	0.369
0.183	1.690	0.826	0.478	0.414	0.466	0.478	0.414
1.239	2.528	1.295	0.758	0.415	0.858	0.758	0.415

1.6 Compute Column and Row Statistics

Compute column and row means, and also column and row sums

```r
print(paste0('colSums=',
             paste(round(
               colSums(mt_rnorm_runif),3), collapse=',')
             ))
```

```
## [1] "colSums=-0.493,4.276,1.859,2.002,1.492,1.962,2.002,1.492"
```

```r
print(paste0('colMeans=',
             paste(round(
               colMeans(mt_rnorm_runif),3), collapse=',')
             ))
```

```
## [1] "colMeans=-0.099,0.855,0.372,0.4,0.298,0.392,0.4,0.298"
```

```r
print(paste0('rowSums=',
             paste(round(
               rowSums(mt_rnorm_runif),3), collapse=',')
             ))
```

```
## [1] "rowSums=1.52,2.329,2.259,3.321,5.163"
```

```r
print(paste0('rowMeans=',
             paste(round(
               rowMeans(mt_rnorm_runif),3), collapse=',')
             ))
```

```
## [1] "rowMeans=0.19,0.291,0.282,0.415,0.645"
```

1.7 Add Column to Matrix with Common Scalar Value

Given some matrix of information, add a column, where all rows of the column have the same numerical value. Use the matrix created prior. - R add column to matrix - r append column to matrix constant value

```
fl_new_first_col_val <- 111
fl_new_last_col_val <- 999
mt_with_more_columns <- cbind(rep(fl_new_first_col_val, dim(mt_rnorm_runif)[1]),
                              mt_rnorm_runif,
                              rep(fl_new_last_col_val, dim(mt_rnorm_runif)[1]))
# Display
kable(mt_with_more_columns) %>% kable_styling_fc_wide()
```

111	-1.6895557	1.2805549	0.5490967	0.4777960	0.1428000	0.1388061	0.4777960	0.1428000	999
111	1.2394959	-1.7272706	0.2382129	0.7584595	0.4145463	0.2330341	0.7584595	0.4145463	999
111	-0.1089660	1.6901844	-1.0488931	0.2164079	0.4137243	0.4659625	0.2164079	0.4137243	999
111	-0.1172420	0.5038124	1.2947633	0.3181810	0.3688455	0.2659726	0.3181810	0.3688455	999
111	0.1830826	2.5283366	0.8255398	0.2316258	0.1524447	0.8578277	0.2316258	0.1524447	999