R Programlama'da Matrisler - Çözümler

Çözümler

S1. Matris Oluşturma (matrix, byrow)

```
v <- 1:6
M_col <- matrix(v, nrow = 2, ncol = 3)           # sütun sütun
M_row <- matrix(v, nrow = 2, ncol = 3, byrow=TRUE) # satır satır
M_col; M_row
```

```
     [,1] [,2] [,3]
[1,]    1    3    5
[2,]    2    4    6
```

```
     [,1] [,2] [,3]
[1,]    1    2    3
[2,]    4    5    6
```

```
dim(M_col); dim(M_row)
```

```
[1] 2 3
```

```
[1] 2 3
```

Açıklama: `matrix()` varsayılan olarak sütun sütun doldurur; `byrow=TRUE` satır satır doldurur.

S2. cbind / rbind ile Matris

```
a <- c(10, 20, 30)
b <- c(40, 50, 60)
MC <- cbind(a, b)  # 3x2
MR <- rbind(a, b)  # 2x3
MC; MR
```

```
      a  b
[1,] 10 40
[2,] 20 50
[3,] 30 60

  [,1] [,2] [,3]
a   10   20   30
b   40   50   60
```

```r
dim(MC); dim(MR)
```

```
[1] 3 2

[1] 2 3
```

S3. İndeksleme ve Mantıksal Seçim

```r
M <- matrix(1:9, nrow = 3, byrow = TRUE)
M
```

```
     [,1] [,2] [,3]
[1,]    1    2    3
[2,]    4    5    6
[3,]    7    8    9
```

```r
elem_23 <- M[2, 3]           # 2. satır 3. sütun
row1    <- M[1, ]            # 1. satır (vektör)
col2    <- M[, 2]            # 2. sütun (vektör)
col2_m  <- M[, 2, drop=FALSE] # 2. sütun (matris)
even_e  <- M[M %% 2 == 0]    # çiftler
list(elem_23=elem_23, row1=row1, col2=col2, dim_col2_m=dim(col2_m), even=even_e)
```

```
$elem_23
[1] 6

$row1
[1] 1 2 3

$col2
[1] 2 5 8

$dim_col2_m
[1] 3 1

$even
[1] 4 2 8 6
```

Not: `drop=FALSE` boyut düşmesini engeller; tek sütun da olsa matris yapısı korunur.

S4. Temel İşlemler ve Transpoz

```r
A <- matrix(c(2,4,6,8), nrow=2)  # 2x2
B <- matrix(c(1,3,5,7), nrow=2)  # 2x2

A_plus_B <- A + B
A_minus_B <- A - B
A_elem_mul <- A * B
A_div_B <- A / B
A_pow2 <- A ^ 2

tA <- t(A)
rS <- rowSums(A)
cM <- colMeans(A)
```

S5. Matris Çarpımı vs Eleman Bazlı Çarpım

```r
X <- matrix(1:6, nrow = 2, byrow = TRUE) # 2x3
Y <- matrix(1:6, nrow = 3, byrow = TRUE) # 3x2
prod_mat <- X %*% Y     # 2x2
dim_prod <- dim(prod_mat)
prod_mat; dim_prod
```

```
     [,1] [,2]
[1,]   22   28
[2,]   49   64
```

```
[1] 2 2
```

Yorum: `%*%` matris çarpımıdır ve iç boyutlar eşit olmalıdır ($2{\times}3$ ile $3{\times}2 \rightarrow 2{\times}2$). `X * Y` ise **eleman bazlı** çarpımdır; boyutlar birebir aynı olmadığından burada anlamsız/hatalıdır.

S6. Birim Matris, Determinant

```r
I3 <- diag(3)
I3
```

```
     [,1] [,2] [,3]
[1,]    1    0    0
[2,]    0    1    0
[3,]    0    0    1
```

```r
D <- matrix(c(2,1,0,
              0,3,0,
              1,0,4), nrow=3, byrow=TRUE)
detD <- det(D)
detD
```

```
[1] 24
```

Kısa bilgi: `det(D)  0` ise `D` terslenebilir. `det(D) = 0` durumunda matris tekildir, ters yoktur.

S7. Ters Matris ve Doğrusal Denklem Çözümü

```r
A <- matrix(c(2, 3,
              1, -1), nrow = 2, byrow = TRUE)
b <- c(8, 1)

A_inv <- solve(A)
x_sol <- solve(A, b)
A_inv; x_sol
```

```
     [,1] [,2]
[1,]  0.2  0.6
[2,]  0.2 -0.4
```

```
[1] 2.2 1.2
```

```r
# Sağlama: A %*% x = b olmalı
A %*% x_sol
```

```
     [,1]
[1,]    8
[2,]    1
```

Açıklama: `solve(A, b)` doğrudan lineer sistemi çözer. Sayısal kararlılık açısından tersi hesaplayıp çarpmaktan genellikle daha güvenlidir.

S8. Mini Uygulama (Not Tablosu)

```r
Not <- matrix(c(70,80,90,
                60,75,85,
                90,95,88), nrow=3, byrow=TRUE)
Not
```

```
     [,1] [,2] [,3]
[1,]   70   80   90
[2,]   60   75   85
[3,]   90   95   88
```

```r
ogr_ort <- rowMeans(Not)     # öğrencilerin ortalaması
snv_max <- apply(Not, 2, max) # sınav bazında maksimum

alt_mat <- Not[2, c(1,3), drop=FALSE]  # 2. öğrencinin 1. ve 3. sınavı (matris)

ogr_ort
```

```
[1] 80.00000 73.33333 91.00000
```

```r
snv_max
```

```
[1] 90 95 90
```

```r
alt_mat
```

```
     [,1] [,2]
[1,]   60   85
```

```r
dim(alt_mat)
```

```
[1] 1 2
```

Not: `drop=FALSE` ile 1×2 yapı korunur; ilerleyen işlemlerde boyut tutarlılığı sağlar.