第 1 章

R で行列を勉強する

1.1 Matrix オブジェクトの作成

1.1.1 行列を作る

```
# 行数と列数を指定する
n <- 2
m <- 3
# 行列の要素数分の値を作る
v <- 1:(n*m)
# 行列を作る
M <- matrix(v,n,m)
M
# 値の詰め方を変える
M2 <- matrix(v,n,m,byrow=TRUE)
M2
```

```
> # 行数と列数を指定する
> n <- 2
> m <- 3
> # 行列の要素数分の値を作る
> v <- 1:(n*m)
> # 行列を作る
> M <- matrix(v,n,m)
> M
     [,1] [,2] [,3]
[1,]    1    3    5
[2,]    2    4    6

> # 値の詰め方を変える
> M2 <- matrix(v,n,m,byrow=TRUE)
```

```
> M2

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

1.1.2　行列の四則演算～数値を納めた箱として扱う

同じ形の行列は数値を納めた箱として扱える。

```
n <- 2
m <- 3
M1 <- matrix(1:(n*m),n,m)
M2 <- matrix(rep(1:2,(n*m)/2),n,m)
M1
M2
M1 + M2
M1 - M2
M1 * M2
M1 / M2
M1 ^ M2
```

```
> n <- 2
> m <- 3
> M1 <- matrix(1:(n*m),n,m)
> M2 <- matrix(rep(1:2,(n*m)/2),n,m)
> M1

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

> M2

     [,1] [,2] [,3]
[1,]    1    1    1
[2,]    2    2    2

> M1 + M2

     [,1] [,2] [,3]
[1,]    2    4    6
[2,]    4    6    8

> M1 - M2
```

```
     [,1] [,2] [,3]
[1,]    0    2    4
[2,]    0    2    4

> M1 * M2

     [,1] [,2] [,3]
[1,]    1    3    5
[2,]    4    8   12

> M1 / M2

     [,1] [,2] [,3]
[1,]    1    3    5
[2,]    1    2    3

> M1 ^ M2

     [,1] [,2] [,3]
[1,]    1    3    5
[2,]    4   16   36
```

1.2 行列の『積』

1.2.1 行列の『いわゆる積』

$n \times m$ 行列 M_1 を $m \times n$ の行列 M_2 に左からかけることができる。行列の積 $M_1 M_2$ は R では

```
M1 %*% M2
```

と書かれこれは、$M1$ の第 i 行ベクトルと $M2$ の第 j 列ベクトルの内積が $M_1 M_2$ の (i, j) 成分となるような計算のこと。

```
n <- 2
m <- 3
M1 <- matrix(1:(n*m),n,m)
M2 <- matrix(rep(1:2,(n*m)/2),m,n)
M1
M2
# 行列の積は次のように書く
M1 %*% M2
```

```
> n <- 2
> m <- 3
> M1 <- matrix(1:(n*m),n,m)
> M2 <- matrix(rep(1:2,(n*m)/2),m,n)
```

```
> M1

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

> M2

     [,1] [,2]
[1,]    1    2
[2,]    2    1
[3,]    1    2

> # 行列の積は次のように書く
> M1 %*% M2

     [,1] [,2]
[1,]   12   15
[2,]   16   20
```

1.2.2　行列の『いろいろな積』

行列の『積』を理解するために 2 つの道を取る。

- 行列をベクトルの束と見て、ベクトルの『積』を考える
- 行列の『積』を『行列の要素の総当たりの積』から考える

ベクトルとベクトルの『積』

長さ n_v のベクトル V と長さ n_u のベクトル U とを考える。これらの要素の総当たりペアをとると、$n_v \times n_u$ 個の値が得られる。これを $n_v \times n_u$ 行列として表現することはごく自然である。(V を列ベクトル、U を行ベクトルとみなしたときの行列の『いわゆる積』ともみなせる)。

$$VU^T = \begin{pmatrix} v_1u_1 & v_1u_2 & \dots & v_1u_{n_u} \\ v_2u_1 & v_2u_2 & \dots & v_2u_{n_u} \\ \dots & \dots & \dots & \dots \\ v_{n_v}u_1 & v_{n_v}u_2 & \dots & v_{n_v}u_{n_u} \end{pmatrix}$$

```
nv <- 3
nu <- 4
V <- 1:nv
U <- 1:nu
V %*% t(U)
```

```
> nv <- 3
> nu <- 4
> V <- 1:nv
```

```
> U <- 1:nu
> V %*% t(U)

     [,1] [,2] [,3] [,4]
[1,]    1    2    3    4
[2,]    2    4    6    8
[3,]    3    6    9   12
```

これを一番「基本的なベクトルとベクトルの積」としよう。
長さが等しい２つのベクトルの場合には、「基本的なベクトルとベクトルの積」は正方行列になる。その対角成分だけを取り出して和を取ったものが、ベクトルの内積

```
> nv <- 3
> nu <- nv
> V <- 1:nv
> U <- 1:nu
> W <- V %*% t(U)
> diag(W)

[1] 1 4 9

> sum(diag(W))

[1] 14

> # 次のように計算してももちろんよい
> sum(V*U)

[1] 14
```

３次元ベクトルの外積

３次元ベクトルの間には「ベクトル積」という『積』があって、これは

$$VU^T = \begin{pmatrix} v_1u_1 & v_1u_2 & v_1u_3 \\ v_2u_1 & v_2u_2 & v_2u_3 \\ v_3u_1 & v_3u_2 & v_3u_3 \end{pmatrix}$$

というベクトルの積の９成分のうち、内積に使われた対角の３成分を使わず、残りの６成分を、対称な２成分のトリオに分け、結果として、３つの数字のベクトルにしたものである。元のベクトルが３次元ベクトルであったが、２つの３次元ベクトルの『積』が３次元ベクトルになっているので、「うまく閉じた関係」になっていることがわかる。この『積』である３次元ベクトルには、有用な性質があることから、名前を「ベクトル積・クロス積」と呼んでいる。

```
my.cross.prod.3d <- function(v,u){
        c(v[2]*u[3]-v[3]*u[2], v[3]*u[1]-v[1]*u[3], v[1]*u[2]-v[2]*u[1])
}
```

行列の要素の総当たりの積

$n_1 \times m_1$ 行列 M_1 と $n_2 \times m_2$ 行列 M_2 との積の計算の仕方は一つではない。M_1 行列には $n_1 \times m_1$ 個の要素があり、M_2 行列には $n_2 \times m_2$ 個の要素があるので、それらの総当たりの積が $(n_1 \times m_1) \times (n_2 \times m_2)$ 個ある。これを使って、複数の『行列の積』が定義できる。この $(n_1 \times m_1) \times (n_2 \times m_2)$ 個の要素を $(n_1 \times m_1)$ 行 $\times \times$ $(n_2 \times m_2)$ 列の行列にするのはクロネッカー積。

```
n1 <- 2
m1 <- 2
n2 <- 3
m2 <-2
M1 <- matrix(1:(n1*n2),n1,m1)
M2 <- matrix(1:(n2*m2),n2,m2)
M1
M2
Kr<- M1 %x% M2
Kr
```

```
> n1 <- 2
> m1 <- 2
> n2 <- 3
> m2 <-2
> M1 <- matrix(1:(n1*n2),n1,m1)
> M2 <- matrix(1:(n2*m2),n2,m2)
> M1
     [,1] [,2]
[1,]    1    3
[2,]    2    4

> M2
     [,1] [,2]
[1,]    1    4
[2,]    2    5
[3,]    3    6

> Kr<- M1 %x% M2
> Kr
     [,1] [,2] [,3] [,4]
[1,]    1    4    3   12
[2,]    2    5    6   15
```

```
[3,]    3    6    9   18
[4,]    2    8    4   16
[5,]    4   10    8   20
[6,]    6   12   12   24
```

$M_1[i_1, j_2] \times M_2[i_2, j_2]$ がクロネッカー積の何行目・何列目になっているかを目視で確認してみるとよい。

$(n_1 \times m_1) \times (n_2 \times m_2)$ 個の要素を $n_1 \times m_1 \times n_2 \times m_2$ と 4 次元の整理箱にしまう (R なら array オブジェクト) のが行列の外積。これは、M_1 の行ベクトルと M_2 の列ベクトルの間のベクトル積を $n_1 \times m_2$ 個求め、それを $n_1 \times m_2$ の行列状に並べたもの。ただし、M_1 の行ベクトルと M_2 の列ベクトルの間のベクトル積自体が $m_1 \times n_2$ 行列なので、出来上がったものは $n_1 \times m_1 \times n_2 \times m_2$ と 4 次元になっている。R であれば以下のようにする。その要素がクロネッカー積の要素と一致することも示しておく (証明ではなく、適当に作ってもいつも要素の値の集合が一致することを示す)。

```
Vpr<- M1 %o% M2
Vpr
range(sort(Kr) - sort(Vpr))
```

```
k <- 100
diffs <- rep(0,k)
for(i in 1:k){
        n1 <- sample(2:6,1)
        m1 <- sample(2:6,1)
        n2 <- sample(2:6,1)
        m2 <- sample(2:6,1)
        M1 <- matrix(runif(n1*m1),n1,m1)
        M2 <- matrix(runif(n2*m2),n2,m2)
        tmp1 <- M1 %o% M2
        tmp2 <- M1 %x% M2
        diffs[i] <- max(abs(sort(c(tmp1))-sort(c(tmp2))))
}
range(diffs)
```

```
> Vpr<- M1 %o% M2
> Vpr

, , 1, 1


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

```
, , 2, 1

     [,1] [,2]
[1,]    2    6
[2,]    4    8

, , 3, 1

     [,1] [,2]
[1,]    3    9
[2,]    6   12

, , 1, 2

     [,1] [,2]
[1,]    4   12
[2,]    8   16

, , 2, 2

     [,1] [,2]
[1,]    5   15
[2,]   10   20

, , 3, 2

     [,1] [,2]
[1,]    6   18
[2,]   12   24

> range(sort(Kr) - sort(Vpr))

[1] 0 0
```

```
> k <- 100
> diffs <- rep(0,k)
> for(i in 1:k){
+         n1 <- sample(2:6,1)
+         m1 <- sample(2:6,1)
+         n2 <- sample(2:6,1)
+         m2 <- sample(2:6,1)
+         M1 <- matrix(runif(n1*m1),n1,m1)
```

```
+         M2 <- matrix(runif(n2*m2),n2,m2)
+         tmp1 <- M1 %o% M2
+         tmp2 <- M1 %x% M2
+         diffs[i] <- max(abs(sort(c(tmp1))-sort(c(tmp2))))
+ }
> range(diffs)

[1] 0 0
```

- $m_1 = n_2$ のとき

1.3 特徴的な行列

1.4 行列の色々

```
# 転置
A <- matrix(1:6,2,3)
t(A)
# 対角成分を取り出す
B <- matrix(1:16,4,4)
B
diag(B)
# 対角行列
D <- diag(1:4)
D
# 単位行列
D. <- diag(4)
D.
D.. <- diag(rep(1,4))
D..
```

連立方程式を解く。

$$Ax = b$$

```
# 連立一次方程式を解く
A <- matrix(sample(1:9),3,3)
b <- c(2,3,5)
result <- solve(A,b)
result
A %*% result
# 逆行列
F <- matrix(rnorm(9),3,3)
```

```
F.inv <- solve(F)
F %*% F.inv
F.inv %*% F
# 固有値分解
eigen.out <- eigen(F)
V <- eigen.out$vectors
S <- diag(eigen.out$values)
V %*% S %*% solve(V)
V%*%t(V)
det(Re(V))
# 対称行列の固有値分解
F. <- F + t(F)
F.
eigen.out.2 <- eigen(F.)
V <- eigen.out.2$vectors
S <- diag(eigen.out.2$values)
V %*% S %*% t(V)
# Vは回転行列
V%*%t(V)
det(Re(V))
# 特異値分解
G <- matrix(rnorm(12),3,4)
G
svd.out <- svd(G)
svd.out$v %*% diag(svd.out$d) %*% svd.out$u
# svd.out$u は回転行列
svd.out$u %*% t(svd.out$u)
```

```
> # 転置
> A <- matrix(1:6,2,3)
> t(A)

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

> # 対角成分を取り出す
> B <- matrix(1:16,4,4)
> B
```

```
     [,1] [,2] [,3] [,4]
[1,]    1    5    9   13
[2,]    2    6   10   14
[3,]    3    7   11   15
[4,]    4    8   12   16

> diag(B)

[1]  1  6 11 16

> # 対角行列
> D <- diag(1:4)
> D

     [,1] [,2] [,3] [,4]
[1,]    1    0    0    0
[2,]    0    2    0    0
[3,]    0    0    3    0
[4,]    0    0    0    4

> # 単位行列
> D. <- diag(4)
> D.

     [,1] [,2] [,3] [,4]
[1,]    1    0    0    0
[2,]    0    1    0    0
[3,]    0    0    1    0
[4,]    0    0    0    1

> D.. <- diag(rep(1,4))
> D..

     [,1] [,2] [,3] [,4]
[1,]    1    0    0    0
[2,]    0    1    0    0
[3,]    0    0    1    0
[4,]    0    0    0    1

> # 連立一次方程式を解く
> A <- matrix(sample(1:9),3,3)
> b <- c(2,3,5)
> result <- solve(A,b)
> result

[1]  2.59574468  0.08510638 -2.70212766
```

```
> A %*% result

     [,1]
[1,]    2
[2,]    3
[3,]    5

> # 逆行列
> F <- matrix(rnorm(9),3,3)
> F.inv <- solve(F)
> F %*% F.inv

              [,1]          [,2]          [,3]
[1,]  1.000000e+00 0.000000e+00 -1.734723e-18
[2,] -4.440892e-16 1.000000e+00 -2.081668e-17
[3,]  0.000000e+00 3.469447e-18  1.000000e+00

> F.inv %*% F

              [,1]          [,2]          [,3]
[1,]  1.000000e+00  0.000000e+00 -1.110223e-16
[2,] -2.775558e-17  1.000000e+00 -3.469447e-17
[3,] -1.734723e-18 -1.301043e-17  1.000000e+00

> # 固有値分解
> eigen.out <- eigen(F)
> V <- eigen.out$vectors
> S <- diag(eigen.out$values)
> V %*% S %*% solve(V)

                [,1]          [,2]          [,3]
[1,] 0.00702074-0i  0.1172555+0i -1.0159939-0i
[2,] 0.23886725+0i  1.4772128+0i  2.8057729+0i
[3,] 1.27285651+0i -0.8744764-0i -0.5092061+0i

> V%*%t(V)

                 [,1]          [,2]          [,3]
[1,]  0.14833904+0i -0.05300655+0i  0.3469751+0i
[2,] -0.05300655+0i  1.85916034+0i -0.7018153+0i
[3,]  0.34697508+0i -0.70181526+0i -0.1361634+0i

> det(Re(V))

[1] 0
```

```
> # 対称行列の固有値分解
> F. <- F + t(F)
> F.

           [,1]       [,2]       [,3]
[1,] 0.01404148 0.3561228  0.2568626
[2,] 0.35612277 2.9544255  1.9312965
[3,] 0.25686264 1.9312965 -1.0184123

> eigen.out.2 <- eigen(F.)
> V <- eigen.out.2$vectors
> S <- diag(eigen.out.2$values)
> V %*% S %*% t(V)

           [,1]       [,2]       [,3]
[1,] 0.01404148 0.3561228  0.2568626
[2,] 0.35612277 2.9544255  1.9312965
[3,] 0.25686264 1.9312965 -1.0184123

> # V は回転行列
> V%*%t(V)

             [,1]         [,2]         [,3]
[1,] 1.000000e+00 3.469447e-18 2.081668e-17
[2,] 3.469447e-18 1.000000e+00 5.551115e-17
[3,] 2.081668e-17 5.551115e-17 1.000000e+00

> det(Re(V))

[1] 1

> # 特異値分解
> G <- matrix(rnorm(12),3,4)
> G

           [,1]       [,2]       [,3]      [,4]
[1,] -0.3712984 -0.7441499 -0.2607415 0.6956332
[2,] -0.3698086  0.6491854 -1.1257994 0.1723844
[3,] -0.1596025 -1.6055589  1.7383332 0.1028702

> svd.out <- svd(G)
> svd.out$v %*% diag(svd.out$d) %*% svd.out$u

            [,1]       [,2]       [,3]
[1,]  0.425782885 -0.1930063  0.2855720
[2,]  0.006907542 -1.8684267 -0.2489062
[3,]  0.457334849  1.6179802  1.2370116
```

```
[4,] -0.242712721   0.3801134 -0.5663997

> # svd.out$u は回転行列
> svd.out$u %*% t(svd.out$u)

     [,1]          [,2]          [,3]
[1,]    1  0.000000e+00  0.000000e+00
[2,]    0  1.000000e+00 -1.665335e-16
[3,]    0 -1.665335e-16  1.000000e+00
```