# データ分析・データ処理 A(多次元データ処理)

HI4 45 号 山口惺司

実験日 2024/05/15

2024/05/22

2024/05/29

# 1. 実験目的

重要なデータ構造の 1 つである高階テンソルの概要が理解でき、R による高階テンソルの処理に関するプログラミング演習を通して、テンソルデータの生成及び主要なテンソルデータ処理ができるようになる。

# 2. 課題 A

1. 課題 1

   [Ex.1-1]

   rTensor のインストール

2. 課題 2

   [Ex.2-1]

   R を使って 3 階テンソルを生成してみよう.

   スクリプト：

```
library(rTensor)
A <- array(0, dim = c(3, 3, 3))
A[1, 1, 1] <- A[2, 2, 1] <- A[3, 3, 1] <- 1
A[2, 1, 2] <- A[3, 2, 2] <- A[1, 3, 2] <- 1
A[3, 1, 3] <- A[1, 2, 3] <- A[2, 3, 3] <- 1
A <- as.tensor((A))
```

   実行結果：

   3 階テンソルを生成することができた.

   説明：

   それぞれの行列の要素に 1 を代入している,

   [Ex.2-2]

   生成した 3 階テンソルの中身を見てみよう.

   スクリプト：

```
A
```

   実行結果：

   実行結果を図 1 に示す.

```
> A
Numeric Tensor of 3 Modes
Modes:  3 3 3
Data:
, , 1

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

, , 2

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

, , 3

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

図 1 Ex2-1 実行結果

説明：

Ex2-1 で作成した 3 階テンソルを出力している.

[Ex.2-3]

R を使って各方向から見た透視図を作ってみよう.

スクリプト：

```
A_1sum <- modeSum(A, 1, drop=TRUE)
A_2sum <- modeSum(A, 2, drop=TRUE)
A_3sum <- modeSum(A, 3, drop=TRUE)
```

実行結果：

各方向から見た透視図を作成できた。

説明：

A_1sum は上から下方向を見た透視図, A_2sum は左から右方向を見た透視図, A_2sum は前から後方向を見た透視図である.

[Ex.2-4]

作成された透視図(行列)を見てみよう.

スクリプト：

```
A1_sum
A2_sum
```

A3_sum

実行結果：

実行結果を図 2 に示す.

```
> print(A_1sum)
Numeric Tensor of 2 Modes
Modes:  3 3
Data:
     [,1] [,2] [,3]
[1,]    1    1    1
[2,]    1    1    1
[3,]    1    1    1
```

```
> print(A_2sum)
Numeric Tensor of 2 Modes
Modes:  3 3
Data:
     [,1] [,2] [,3]
[1,]    1    1    1
[2,]    1    1    1
[3,]    1    1    1
```

```
> print(A_3sum)
Numeric Tensor of 2 Modes
Modes:  3 3
Data:
     [,1] [,2] [,3]
[1,]    1    1    1
[2,]    1    1    1
[3,]    1    1    1
```

図 2 Ex2-3 実行結果

説明：

Ex2-3 で作成した透視図を出力している.

3. 課題 3

[Ex.3-1]

電球がセットされている位置のマップを 1-モード行列展開で作ってみよう.

スクリプト：

```
A1 <- unfold(A, row_idx = 1, col_idx = c(3, 2))
A1
```

実行結果：

実行結果を図 3 に示す.

```
> A1
Numeric Tensor of 2 Modes
Modes:  3 9
Data:
     [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9]
[1,]    1    0    0    0    0    1    0    1    0
[2,]    0    1    0    1    0    0    0    0    1
[3,]    0    0    1    0    1    0    1    0    0
```

図 3 Ex3-1 実行結果

説明：

1-モード行列展開により A を展開している.

[Ex.3-2]

電球がセットされている位置のマップを 2-モード行列展開で作ってみよう.

スクリプト：

```
A2 <- unfold(A, row_idx = 2, col_idx = c(1, 3))
A2
```

実行結果：

実行結果を図 4 に示す.

```
> A2
Numeric Tensor of 2 Modes
Modes:  3 9
Data:
     [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9]
[1,]    1    0    0    0    1    0    0    0    1
[2,]    0    1    0    0    0    1    1    0    0
[3,]    0    0    1    1    0    0    0    1    0
```

図 4 Ex3-2 実行結果

説明：

2-モード行列展開により A を展開している.

[Ex.3-3]

電球がセットされている位置のマップを 3-モード行列展開で作ってみよう.

スクリプト：

```
A3 <- unfold(A, row_idx = 3, col_idx = c(2, 1))
A3
```

実行結果：

実行結果を図 4 に示す.

```
> A3
Numeric Tensor of 2 Modes
Modes:  3 9
Data:
     [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9]
[1,]    1    0    0    0    1    0    0    0    1
[2,]    0    0    1    1    0    0    0    1    0
[3,]    0    1    0    0    0    1    1    0    0
```

図 5 Ex3-3 実行結果

説明：

3-モード行列展開により A を展開している.

4. 課題 4

[Ex.4-1]

1-モード行列展開で作られたマップを元の立方体に戻してみよう

スクリプト：

```
A_1org <- fold(A1, row_idx = 1, col_idx = c(3, 2), modes = c(3, 3, 3))
```

A_1org

実行結果：

実行結果を図 6 に示す.

```
> A_1org
Numeric Tensor of 3 Modes
Modes:  3 3 3
Data:
, , 1

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

, , 2

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

, , 3

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

図 6 Ex4-1 実行結果

説明：

Ex.3-1 で作成したマップを元の立体に戻している.

[Ex.4-2]

2-モード行列展開で作られたマップを元の立方体に戻してみよう

スクリプト：

A_2org <- fold(A1, row_idx = 2, col_idx = c(1, 3), modes = c(3, 3, 3))

A_2org

実行結果：

実行結果を図 7 に示す.

```
> A_2org
Numeric Tensor of 3 Modes
Modes:  3 3 3
Data:
, , 1

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

, , 2

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

, , 3

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

図 7 Ex4-2 実行結果

説明：

Ex.3-2 で作成したマップを元の立体に戻している.

[Ex.4-3]

3-モード行列展開で作られたマップを元の立方体に戻してみよう

スクリプト：

```
A_3org <- fold(A1, row_idx = 3, col_idx = c(2, 1), modes = c(3, 3, 3))
print(A_3org)
```

実行結果：

実行結果を図 8 に示す.

```
> A_3org
Numeric Tensor of 3 Modes
Modes:  3 3 3
Data:
, , 1

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

, , 2

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

, , 3

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

図 8 Ex4-3 実行結果

説明：

Ex.3-3 で作成したマップを元の立体に戻している．

5. 課題 5

[基本問題 A]

1～24 の値を持つ 3 階テンソル に対して，それぞれ 3 つの 行列の 1～3-モード積を計算した結果を rTensor パッケージ を利用して求めなさい．

スクリプト：

```
library(rTensor)

A <- array(1:24, dim=c(2, 3, 4))
A <- as.tensor(A)
(A@data)

U1 <- array(1:4, dim=c(2, 2))
U2 <- array(1:9, dim=c(3, 3))
U3 <- array(1:16, dim=c(4, 4))

ansA1 <- ttm(A, U1, m=1)
ansA2 <- ttm(A, U2, m=2)
ansA3 <- ttm(A, U3, m=3)

ansA1
```

ansA2

ansA3

実行結果：

実行結果を図 9 に示す.

```
> ansA1
Numeric Tensor of 3 Modes
Modes:  2 3 4
Data:
, , 1

     [,1] [,2] [,3]
[1,]    7   15   23
[2,]   10   22   34

, , 2

     [,1] [,2] [,3]
[1,]   31   39   47
[2,]   46   58   70

, , 3

     [,1] [,2] [,3]
[1,]   55   63   71
[2,]   82   94  106

, , 4

     [,1] [,2] [,3]
[1,]   79   87   95
[2,]  118  130  142
```

```
> ansA2
Numeric Tensor of 3 Modes
Modes:  2 3 4
Data:
, , 1

     [,1] [,2] [,3]
[1,]   48   57   66
[2,]   60   72   84

, , 2

     [,1] [,2] [,3]
[1,]  120  147  174
[2,]  132  162  192

, , 3

     [,1] [,2] [,3]
[1,]  192  237  282
[2,]  204  252  300

, , 4

     [,1] [,2] [,3]
[1,]  264  327  390
[2,]  276  342  408
```

```
> ansA3
Numeric Tensor of 3 Modes
Modes:  2 3 4
Data:
, , 1

     [,1] [,2] [,3]
[1,]  400  456  512
[2,]  428  484  540

, , 2

     [,1] [,2] [,3]
[1,]  440  504  568
[2,]  472  536  600

, , 3

     [,1] [,2] [,3]
[1,]  480  552  624
[2,]  516  588  660

, , 4

     [,1] [,2] [,3]
[1,]  520  600  680
[2,]  560  640  720
```

図 9 Ex.5-基本課題 A 実行結果

説明：

ttm()関数を用いて、3 階テンソル A に対して、行列 U1~U3 のモード積を計算している.

[応用課題 A]

例題の 3 階テンソルと行列を用いて簡易的な求め方の n-モード積を計算するスクリプトを作成してください.

スクリプト：

```
library(rTensor)

A <- array(1:24, dim=c(2, 3, 4))
A <- as.tensor(A)
(A@data)

U1 <- array(1:4, dim=c(2, 2))
U2 <- array(1:9, dim=c(3, 3))
```

```
U3 <- array(1:16, dim=c(4, 4))

A1 <- unfold(A, row_idx=1, col_idx=c(2, 3))
A2 <- unfold(A, row_idx=2, col_idx=c(1, 3))
A3 <- unfold(A, row_idx=3, col_idx=c(1, 2))

idx1 <- U1 %*% A1@data
idx2 <- U2 %*% A2@data
idx3 <- U3 %*% A3@data

myA1 <- fold(idx1, row_idx=1, col_idx=c(2, 3), modes=c(2, 3, 4))
myA2 <- fold(idx2, row_idx=2, col_idx=c(1, 3), modes=c(2, 3, 4))
myA3 <- fold(idx3, row_idx=3, col_idx=c(1, 2), modes=c(2, 3, 4))

myA1
myA2
myA3
```

実行結果：

実行結果を図 10 に示す.

```
> myA1
Numeric Tensor of 3 Modes
Modes:  2 3 4
Data:
, , 1

     [,1] [,2] [,3]
[1,]    7   15   23
[2,]   10   22   34

, , 2

     [,1] [,2] [,3]
[1,]   31   39   47
[2,]   46   58   70

, , 3

     [,1] [,2] [,3]
[1,]   55   63   71
[2,]   82   94  106

, , 4

     [,1] [,2] [,3]
[1,]   79   87   95
[2,]  118  130  142
```

```
> myA2
Numeric Tensor of 3 Modes
Modes:  2 3 4
Data:
, , 1

     [,1] [,2] [,3]
[1,]   48   57   66
[2,]   60   72   84

, , 2

     [,1] [,2] [,3]
[1,]  120  147  174
[2,]  132  162  192

, , 3

     [,1] [,2] [,3]
[1,]  192  237  282
[2,]  204  252  300

, , 4

     [,1] [,2] [,3]
[1,]  264  327  390
[2,]  276  342  408
```

```
> myA3
Numeric Tensor of 3 Modes
Modes:  2 3 4
Data:
, , 1

     [,1] [,2] [,3]
[1,]  400  456  512
[2,]  428  484  540

, , 2

     [,1] [,2] [,3]
[1,]  440  504  568
[2,]  472  536  600

, , 3

     [,1] [,2] [,3]
[1,]  480  552  624
[2,]  516  588  660

, , 4

     [,1] [,2] [,3]
[1,]  520  600  680
[2,]  560  640  720
```

図 10 Ex5-応用課題 A 実行結果

説明：

[応用課題 B]

マクマホンキューブと行列との 1〜3-モード積を求めるスクリプトを作成してください． ただし，ttm 関数を用いた場合と簡易的な求め方での場合の 2 種類を実装してください．

スクリプト：

```
library(rTensor)

A <- array(0, dim=c(3, 3, 3))
A <- as.tensor(A)
(A@data)

A[2, 2, 1] <- 3
A[1, 2, 2] <- 1
A[2, 3, 2] <- 2
A[2, 1, 2] <- 4
A[3, 2, 2] <- 6
A[2, 2, 3] <- 5

U <- array(0, dim=c(3, 3))
U[3, 1] <- U[2, 2] <- U[1, 3] <- 1

ansA1 <- ttm(A, U, m=1)
ansA2 <- ttm(A, U, m=2)
ansA3 <- ttm(A, U, m=3)
ansA1
ansA2
ansA3

A1 <- unfold(A, row_idx=1, col_idx=c(2, 3))
A2 <- unfold(A, row_idx=2, col_idx=c(1, 3))
A3 <- unfold(A, row_idx=3, col_idx=c(1, 2))

idx1 <- U %*% A1@data
idx2 <- U %*% A2@data
idx3 <- U %*% A3@data

myA1 <- fold(idx1, row_idx=1, col_idx=c(2, 3), modes=c(3, 3, 3))
myA2 <- fold(idx2, row_idx=2, col_idx=c(1, 3), modes=c(3, 3, 3))
myA3 <- fold(idx3, row_idx=3, col_idx=c(1, 2), modes=c(3, 3, 3))
myA1
myA2
myA3
```

実行結果：

ttm()関数を用いた場合の実行結果を図 11、簡易的な求め方での場合の実行結果を図 12 に示す.

```
> ansA1
Numeric Tensor of 3 Modes
Modes:  2 3 4
Data:
, , 1

     [,1] [,2] [,3]
[1,]    7   15   23
[2,]   10   22   34

, , 2

     [,1] [,2] [,3]
[1,]   31   39   47
[2,]   46   58   70

, , 3

     [,1] [,2] [,3]
[1,]   55   63   71
[2,]   82   94  106

, , 4

     [,1] [,2] [,3]
[1,]   79   87   95
[2,]  118  130  142
```

```
> ansA2
Numeric Tensor of 3 Modes
Modes:  2 3 4
Data:
, , 1

     [,1] [,2] [,3]
[1,]   48   57   66
[2,]   60   72   84

, , 2

     [,1] [,2] [,3]
[1,]  120  147  174
[2,]  132  162  192

, , 3

     [,1] [,2] [,3]
[1,]  192  237  282
[2,]  204  252  300

, , 4

     [,1] [,2] [,3]
[1,]  264  327  390
[2,]  276  342  408
```

```
> ansA3
Numeric Tensor of 3 Modes
Modes:  2 3 4
Data:
, , 1

     [,1] [,2] [,3]
[1,]  400  456  512
[2,]  428  484  540

, , 2

     [,1] [,2] [,3]
[1,]  440  504  568
[2,]  472  536  600

, , 3

     [,1] [,2] [,3]
[1,]  480  552  624
[2,]  516  588  660

, , 4

     [,1] [,2] [,3]
[1,]  520  600  680
[2,]  560  640  720
```

図 11 Ex5-応用課題 B-ttm()関数を用いた場合の実行結果

```
> myA1
Numeric Tensor of 3 Modes
Modes:  2 3 4
Data:
, , 1

     [,1] [,2] [,3]
[1,]    7   15   23
[2,]   10   22   34

, , 2

     [,1] [,2] [,3]
[1,]   31   39   47
[2,]   46   58   70

, , 3

     [,1] [,2] [,3]
[1,]   55   63   71
[2,]   82   94  106

, , 4

     [,1] [,2] [,3]
[1,]   79   87   95
[2,]  118  130  142
```

```
> myA2
Numeric Tensor of 3 Modes
Modes:  2 3 4
Data:
, , 1

     [,1] [,2] [,3]
[1,]   48   57   66
[2,]   60   72   84

, , 2

     [,1] [,2] [,3]
[1,]  120  147  174
[2,]  132  162  192

, , 3

     [,1] [,2] [,3]
[1,]  192  237  282
[2,]  204  252  300

, , 4

     [,1] [,2] [,3]
[1,]  264  327  390
[2,]  276  342  408
```

```
> myA3
Numeric Tensor of 3 Modes
Modes:  2 3 4
Data:
, , 1

     [,1] [,2] [,3]
[1,]  400  456  512
[2,]  428  484  540

, , 2

     [,1] [,2] [,3]
[1,]  440  504  568
[2,]  472  536  600

, , 3

     [,1] [,2] [,3]
[1,]  480  552  624
[2,]  516  588  660

, , 4

     [,1] [,2] [,3]
[1,]  520  600  680
[2,]  560  640  720
```

図 12 Ex5-応用課題 B-簡易的な求め方での場合の実行結果

説明：

ttm()関数を使用してマクマホンキューブと行列との 1～3-モード積を求めている。

簡易的な求め方では unfold()関数でテンソルを展開し、積を計算した後に fold()関数でテンソルを畳み込んでいる。

6. 課題 6

[Ex6-1]

関数 eigen の使用例

スクリプト：

```
x <- c(1, 0, 0, 0, 3, -1, 0, -1, 3)
A <- array(x, dim=c(3, 3))
z <- eigen(A)
lambda <- z$values
T <- z$vectors
lambda
T
```

実行結果：

実行結果を図 13 に示す.

```
> lambda
[1] 4 2 1
> T
           [,1]      [,2] [,3]
[1,]  0.0000000 0.0000000    1
[2,] -0.7071068 0.7071068    0
[3,]  0.7071068 0.7071068    0
```

図 13 Ex6-1 実行結果

説明：

eigen()関数を使用して、固有値と固有ベクトルの計算を行い、A の固有値の組と A の固有行列を出力している。

[Ex6-2]

関数 svd の使用例

スクリプト：

```
x <- c(1, 1, 2, 2, 2, -1, 0, -1)
B <- array(x, dim=c(4, 2))
z <- svd(B)
sigma <- z$d
U <- z$u
```

V <- z$v

sigma

V

実行結果：

実行結果を図 14 に示す.

```
> sigma
[1] 3.199386 2.400819
> V
           [,1]      [,2]
[1,] -0.9732490 0.2297529
[2,]  0.2297529 0.9732490
> U
           [,1]       [,2]
[1,] -0.1605756  0.9064619
[2,] -0.3760103 -0.3096843
[3,] -0.6083974  0.1913955
[4,] -0.6802090 -0.2139866
```

図 14 Ex6-2 実行結果

説明：

svd()関数で B の特異値と特異行列を計算し、B の右特異行列と左特異行列を出力している。

[Ex6-3]

[Ex.6-1]で得られた lamda, T を利用して、スライド p.4 の式の右辺の計算を実装し、左辺の元データ A に戻ることを確認せよ。

スクリプト：

A1 <- T %*% diag(lambda) %*% t(T)

A1

実行結果：

実行結果を図 15 に示す.

```
> A1
     [,1] [,2] [,3]
[1,]    1    0    0
[2,]    0    3   -1
[3,]    0   -1    3
```

図 15 Ex6-3 実行結果

説明：

$A=T\Lambda T^T$ の右辺の計算をして元データの行列を求めている。

[Ex6-4]

[Ex.6-2]で得られた sigma, U, V を利用して、スライド p.9 の式の右辺の 計算を実装し、左辺の元データ B に戻ることを確認せよ。

スクリプト：

```
B1 <- U %*% diag(sigma) %*% t(V)
B1
```

実行結果：

実行結果を図 16 に示す.

```
> B1
     [,1]          [,2]
[1,]    1  2.000000e+00
[2,]    1 -1.000000e+00
[3,]    2 -2.220446e-16
[4,]    2 -1.000000e+00
```

図 16 Ex6-4 実行結果

説明：

$B=U\Sigma V^T$ の右辺を計算して元データの行列を求めている.

7. 課題 7

[Ex7-1]

R 成分画像データ SVD(512 個の特異値に分解したとき)

スクリプト：

```
library(imager)
img <- load.image("parrots.png")
plot(img)
A <- t(img[ , , 1, 1])

plot(as.cimg(t(A)))
svdA <- svd(A)
sigma <- svdA$d
U <- svdA$u
V <- svdA$v
recA <- U %*% diag(sigma) %*% t(V)
plot(as.cimg(t(recA)))
```

実行結果：

実行結果を図 17～19 に示す.

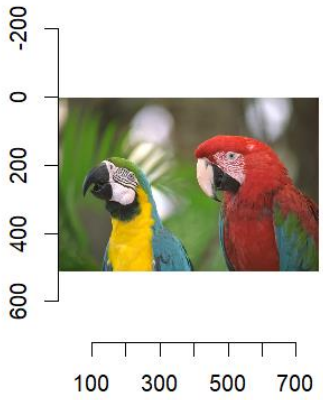


図 17 元画像

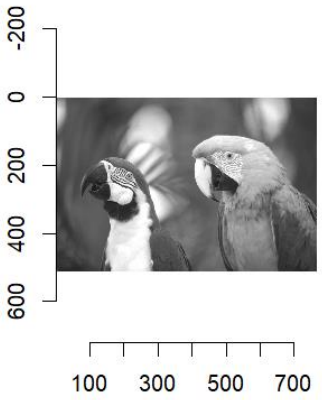


図 18 R 成分画像

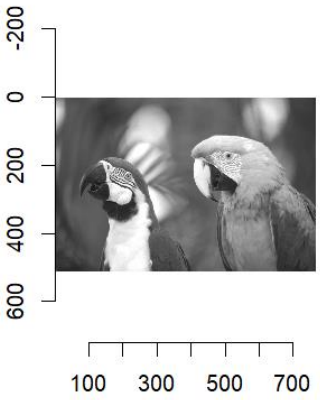


図 19 再構成画像

説明：

特異値 512 個の時の元画像 R 成分画像、再構成画像を出力

[Ex7-2]

R 成分画像データの SVD(10 個の特異値に分解したとき)

スクリプト：

```
num_sv <- 10
svdA2 <- svd(A, nu=num_sv, nv=num_sv)
sigma2 <- svdA2$d
U2 <- svdA2$u
V2 <- svdA2$v
recA2 <- U2 %*% diag(sigma2[1:num_sv]) %*% t(V2)
plot(as.cimg(t(recA2)))

library(rTensor)
w <- width(img); h <- height(img); MAX <- 1.0
MSE_R <- fnorm(as.tensor(A-recA2))^2 / (h*w)
PSNR_R <- 10*log10(MAX^2/MSE_R)
n <- h; p <- w; r <- 10; c_raito <- 1-(n*r+r*r+p*r) / (n*p)
```

実行結果：

実行結果を図 20、図 21 に示す。

```
> PSNR_R
[1] 24.44305
> c_raito
[1] 0.9671936
```

図 20 計算した R 成分画像の PSNR と SVD の圧縮率

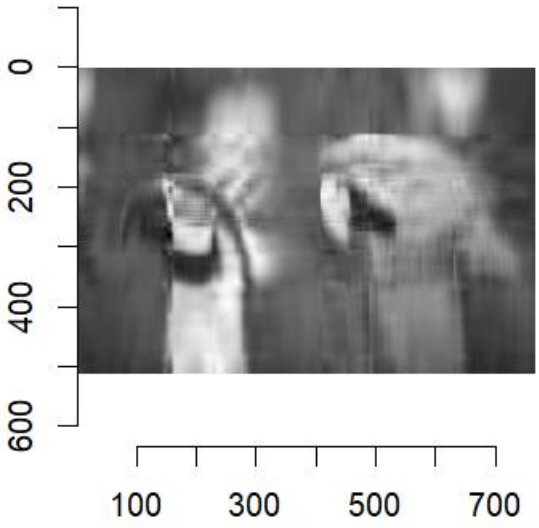


図 21 R 成分画像データの SVD(10 個の特異値に分解)

説明：

分解する特異値の数を 10 個にして SVD を出力。

[Ex7-3]

各自、RG 画像を収集し、その G 成分の画像 SVD を計算してください。なお、PSNR の値が約 25dB, 30dB, 40dB の 3 つについて、分解する特異値の数を調整して、計算してください.

スクリプト：

```
library(imager)
library(rTensor)

img <- load.image("mogusi2.png")
plot(img)
A <- t(img[ , , 1, 2])

num_sv <- 65
#num_sv <- 130
#num_sv <- 240
svdA <- svd(A, nu=num_sv, nv=num_sv)
sigma <- svdA$d
U <- svdA$u
V <- svdA$v
recA <- U %*% diag(sigma[1:num_sv]) %*% t(V)
plot(as.cimg(t(recA2)))
```

```
w <- width(img); h <- height(img); MAX <- 1.0
MSE_G <- fnorm(as.tensor(A-recA))^2 / (h*w)
PSNR_G <- 10*log10(MAX^2/MSE_G)
n <- h; p <- w; r <- num_sv; c_raito <- 1-(n*r+r*r+p*r) / (n*p)

num_sv
PSNR_G
c_raito
```

実行結果：

実行結果を図 22,23 に示す。

```
> num_sv
[1] 65
> PSNR_G
[1] 25.26441
> c_raito
[1] 0.7115625

> num_sv
[1] 130
> PSNR_G
[1] 30.35767
> c_raito
[1] 0.3879167

> num_sv
[1] 240
> PSNR_G
[1] 40.09356
> c_raito
[1] -0.24
```

図 22 分解する特異値、G 成分の PSNR 値、SVD の圧縮率

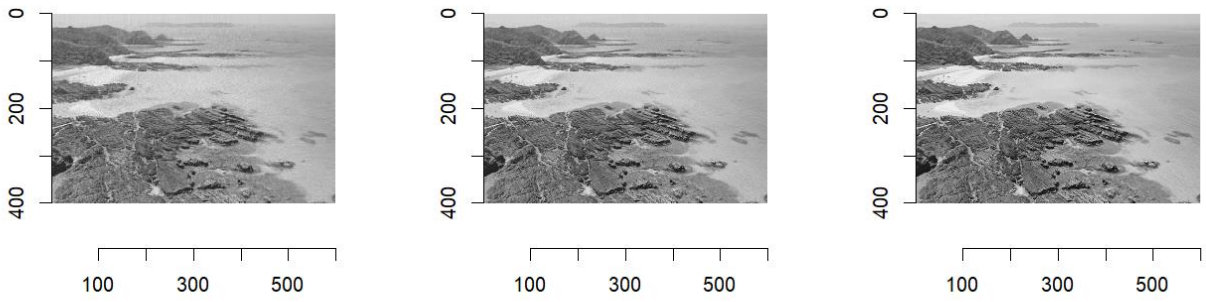


図 23 PSNR 値 20dB, 25dB, 40dB で出力した画像

説明：

G 成分の PSNR 値がそれぞれ 25dB,30dB,40dB に近づくように分解する特異値の数を調整したらそれぞれ 65 個, 130 個, 240 個となった。

## 3. 課題 B

大きさ 4x4x4 の立方体において、以下の各要素の組み合わせで電球をセットして点灯されたとき、立方体の 3 方向から見てすべてが点灯されるものを選択してください。ただし、R スクリプトを作成し、実行して調べて回答してください。

[1, 6, 12, 15, 20, 23, 26, 29, 34, 37, 43, 48, 51, 56, 67, 62]

[1, 8, 11, 14, 20, 23, 26, 29, 35, 38, 41, 48, 50, 53, 60, 63]
[2, 5, 12, 15, 17, 22, 27, 32, 36, 39, 42, 45, 51, 56, 57, 62]
[1, 7, 10, 16, 19, 24, 25, 30, 34, 37, 44, 47, 52, 54, 59, 61]

スクリプト：

```
library(rTensor)
A1 <- array(0, dim = c(4, 4, 4))
A2 <- array(0, dim = c(4, 4, 4))
A3 <- array(0, dim = c(4, 4, 4))
A4 <- array(0, dim = c(4, 4, 4))

x1 <- c(2,5,12,15,17,22,27,32,36,39,42,45,51,56,57,62)
x2 <- c(1,6,12,15,20,23,26,29,34,37,43,48,51,56,57,62)
x3 <- c(1,7,10,16,19,24,25,30,34,37,44,47,52,54,59,61)
x4 <- c(1,8,11,14,20,23,26,29,35,38,41,48,50,53,60,63)

for(i in 1:4){
  for(j in 1:4){
    for(k in 1:4){
      sum = k + (j-1)*4 + (i-1)*16
      if(sum %in% x1){
        A1[k, j, i] <- 1
      }
      if(sum %in% x2){
        A2[k, j, i] <- 1
      }
      if(sum %in% x3){
        A3[k, j, i] <- 1
      }
      if(sum %in% x4){
        A4[k, j, i] <- 1
      }
    }
  }
}

A1 <- as.tensor((A1))
A2 <- as.tensor((A2))
A3 <- as.tensor((A3))
A4 <- as.tensor((A4))

print("A1:")
```

```
print(modeSum(A1, 1, drop=TRUE))
print(modeSum(A1, 2, drop=TRUE))
print(modeSum(A1, 3, drop=TRUE))

print("A2:")
print(modeSum(A2, 1, drop=TRUE))
print(modeSum(A2, 2, drop=TRUE))
print(modeSum(A2, 3, drop=TRUE))

print("A3:")
print(modeSum(A3, 1, drop=TRUE))
print(modeSum(A3, 2, drop=TRUE))
print(modeSum(A3, 3, drop=TRUE))

print("A4:")
print(modeSum(A4, 1, drop=TRUE))
print(modeSum(A4, 2, drop=TRUE))
print(modeSum(A4, 3, drop=TRUE))
```

実行結果：

実行結果を図 24 に示す。

```
[1] "A1:"
Numeric Tensor of 2 Modes
Modes:  4 4
Data:
     [,1] [,2] [,3] [,4]
[1,]    1    1    1    1
[2,]    1    1    1    1
[3,]    1    1    1    1
[4,]    1    1    1    1
Numeric Tensor of 2 Modes
Modes:  4 4
Data:
     [,1] [,2] [,3] [,4]
[1,]    1    1    1    1
[2,]    1    1    1    1
[3,]    1    1    1    1
[4,]    1    1    1    1
Numeric Tensor of 2 Modes
Modes:  4 4
Data:
     [,1] [,2] [,3] [,4]
[1,]    1    1    1    1
[2,]    1    1    1    1
[3,]    1    1    1    1
[4,]    1    1    1    1
[1] "A2:"
Numeric Tensor of 2 Modes
Modes:  4 4
Data:
     [,1] [,2] [,3] [,4]
[1,]    1    1    1    1
[2,]    1    1    1    1
[3,]    1    1    1    1
[4,]    1    1    1    1
Numeric Tensor of 2 Modes
Modes:  4 4
Data:
     [,1] [,2] [,3] [,4]
[1,]    1    1    1    1
[2,]    1    1    1    1
[3,]    1    1    1    1
[4,]    1    1    1    1
Numeric Tensor of 2 Modes
Modes:  4 4
Data:
     [,1] [,2] [,3] [,4]
[1,]    1    1    1    1
[2,]    1    1    1    1
[3,]    1    1    1    1
[4,]    1    1    1    1
[1] "A3:"
Numeric Tensor of 2 Modes
Modes:  4 4
Data:
     [,1] [,2] [,3] [,4]
[1,]    1    1    1    1
[2,]    1    1    1    1
[3,]    1    1    1    1
[4,]    1    1    1    1
Numeric Tensor of 2 Modes
Modes:  4 4
Data:
     [,1] [,2] [,3] [,4]
[1,]    1    1    1    1
[2,]    1    1    1    1
[3,]    1    1    1    1
[4,]    1    1    1    1
Numeric Tensor of 2 Modes
Modes:  4 4
Data:
     [,1] [,2] [,3] [,4]
[1,]    1    1    1    1
[2,]    1    1    1    1
[3,]    1    1    1    1
[4,]    1    1    1    1
[1] "A4:"
Numeric Tensor of 2 Modes
Modes:  4 4
Data:
     [,1] [,2] [,3] [,4]
[1,]    1    1    1    1
[2,]    1    1    1    1
[3,]    1    1    1    1
[4,]    1    1    1    1
Numeric Tensor of 2 Modes
Modes:  4 4
Data:
     [,1] [,2] [,3] [,4]
[1,]    1    1    1    1
[2,]    1    1    1    1
[3,]    1    1    1    1
[4,]    1    1    1    1
Numeric Tensor of 2 Modes
Modes:  4 4
Data:
     [,1] [,2] [,3] [,4]
[1,]    1    1    1    1
[2,]    1    1    1    1
[3,]    1    1    1    1
[4,]    1    1    1    1
```

図 24 各立方体を各方向から見たときの値

説明：

3 重の for 文を用いて与えられた配列を 4x4x4 テンソルに格納して、各方向から見た透視図を作成した。

出力された透視図を見ると、全て 1 となっているため、大きさ 4x4x4 の立方体において、各要素の組み合わせで電球をセットして点灯されたとき、すべての立方体が 3 方向から見てすべてが点灯される。

## 4. 感想

今までいろんなプログラミング言語を学んできたが、多次元配列を用いる時はRがとても便利だということを実感できた。
課題Bではfor文を用いて自動的に4x4x4のテンソルに値を入れることができてよかった。