데이터 사이언스 과제4

< 1. 선형대수와 R >

```
> A <- matrix(data = 1:36, nrow = 6)
> A
     [,1] [,2] [,3] [,4] [,5] [,6]
[1,]    1    7   13   19   25   31
[2,]    2    8   14   20   26   32
[3,]    3    9   15   21   27   33
[4,]    4   10   16   22   28   34
[5,]    5   11   17   23   29   35
[6,]    6   12   18   24   30   36
> B <- matrix(data = 1:30, nrow = 6)
> B
     [,1] [,2] [,3] [,4] [,5]
[1,]    1    7   13   19   25
[2,]    2    8   14   20   26
[3,]    3    9   15   21   27
[4,]    4   10   16   22   28
[5,]    5   11   17   23   29
[6,]    6   12   18   24   30
> A %*% B
     [,1] [,2] [,3] [,4] [,5]
[1,]  441 1017 1593 2169 2745
[2,]  462 1074 1686 2298 2910
[3,]  483 1131 1779 2427 3075
[4,]  504 1188 1872 2556 3240
[5,]  525 1245 1965 2685 3405
[6,]  546 1302 2058 2814 3570
>
> A <- matrix(data = 1:36, nrow = 6)
> A
     [,1] [,2] [,3] [,4] [,5] [,6]
[1,]    1    7   13   19   25   31
[2,]    2    8   14   20   26   32
[3,]    3    9   15   21   27   33
[4,]    4   10   16   22   28   34
[5,]    5   11   17   23   29   35
[6,]    6   12   18   24   30   36
> B <- matrix(data = 11:46, nrow = 6)
> B
     [,1] [,2] [,3] [,4] [,5] [,6]
[1,]   11   17   23   29   35   41
[2,]   12   18   24   30   36   42
[3,]   13   19   25   31   37   43
[4,]   14   20   26   32   38   44
[5,]   15   21   27   33   39   45
[6,]   16   22   28   34   40   46
> A * B
     [,1] [,2] [,3] [,4] [,5] [,6]
[1,]   11  119  299  551  875 1271
[2,]   24  144  336  600  936 1344
[3,]   39  171  375  651  999 1419
[4,]   56  200  416  704 1064 1496
[5,]   75  231  459  759 1131 1575
[6,]   96  264  504  816 1200 1656
>
> X <- matrix(data=1:10, nrow= 10)
> X
      [,1]
 [1,]    1
 [2,]    2
 [3,]    3
 [4,]    4
 [5,]    5
 [6,]    6
 [7,]    7
 [8,]    8
 [9,]    9
[10,]   10
> Y <- matrix(data= 11:20, nrow= 10)
> Y
      [,1]
 [1,]   11
 [2,]   12
 [3,]   13
 [4,]   14
 [5,]   15
 [6,]   16
 [7,]   17
 [8,]   18
 [9,]   19
[10,]   20
> dotProduct <- function(X,Y) {
+   as.vector(t(X) %*% Y)
+ }
> dotProduct(X,Y)
[1] 935
>
> A <- matrix (data = 1:25 , nrow = 5)
> B <- matrix (data = 26 :50 , nrow = 5)
> C <- matrix (data = 51 :75 , nrow = 5)
> A %*% (B + C)
     [,1] [,2] [,3] [,4] [,5]
[1,] 4555 5105 5655 6205 6755
[2,] 4960 5560 6160 6760 7360
[3,] 5365 6015 6665 7315 7965
[4,] 5770 6470 7170 7870 8570
[5,] 6175 6925 7675 8425 9175
> A %*% B + A %*% C
     [,1] [,2] [,3] [,4] [,5]
[1,] 4555 5105 5655 6205 6755
```

```
[2,] 4960 5560 6160 6760 7360
[3,] 5365 6015 6665 7315 7965
[4,] 5770 6470 7170 7870 8570
[5,] 6175 6925 7675 8425 9175
>
> A <- matrix (data = 1:25 , nrow = 5)
> B <- matrix (data = 26 :50 , nrow = 5)
> C <- matrix (data = 51 :75 , nrow = 5)
> (A %*% B) %*% C
       [,1]   [,2]   [,3]   [,4]    [,5]
[1,] 569850 623350 676850 730350  783850
[2,] 620450 678700 736950 795200  853450
[3,] 671050 734050 797050 860050  923050
[4,] 721650 789400 857150 924900  992650
[5,] 772250 844750 917250 989750 1062250
> A %*% (B %*% C)
       [,1]   [,2]   [,3]   [,4]    [,5]
[1,] 569850 623350 676850 730350  783850
[2,] 620450 678700 736950 795200  853450
[3,] 671050 734050 797050 860050  923050
[4,] 721650 789400 857150 924900  992650
[5,] 772250 844750 917250 989750 1062250
>
> A <- matrix (data = 1:25 , nrow = 5)
> B <- matrix (data = 26 :50 , nrow = 5)
> A %*% B
     [,1] [,2] [,3] [,4] [,5]
[1,] 1590 1865 2140 2415 2690
[2,] 1730 2030 2330 2630 2930
[3,] 1870 2195 2520 2845 3170
[4,] 2010 2360 2710 3060 3410
[5,] 2150 2525 2900 3275 3650
> B %*% A
     [,1] [,2] [,3] [,4] [,5]
[1,]  590 1490 2390 3290 4190
[2,]  605 1530 2455 3380 4305
[3,]  620 1570 2520 3470 4420
[4,]  635 1610 2585 3560 4535
[5,]  650 1650 2650 3650 4650
>
> A <- matrix (data = 1:25 , nrow = 5, ncol = 5, byrow = TRUE )
> A
     [,1] [,2] [,3] [,4] [,5]
[1,]    1    2    3    4    5
[2,]    6    7    8    9   10
[3,]   11   12   13   14   15
[4,]   16   17   18   19   20
[5,]   21   22   23   24   25
> t(A)
     [,1] [,2] [,3] [,4] [,5]
[1,]    1    6   11   16   21
[2,]    2    7   12   17   22
[3,]    3    8   13   18   23
[4,]    4    9   14   19   24
[5,]    5   10   15   20   25
>
> A <- matrix (data = 1:25 , nrow = 5)
> B <- matrix (data = 25 :49 , nrow = 5)
> t(A %*% B)
     [,1] [,2] [,3] [,4] [,5]
[1,] 1535 1670 1805 1940 2075
[2,] 1810 1970 2130 2290 2450
[3,] 2085 2270 2455 2640 2825
[4,] 2360 2570 2780 2990 3200
[5,] 2635 2870 3105 3340 3575
> t(B) %*% t(A)
     [,1] [,2] [,3] [,4] [,5]
[1,] 1535 1670 1805 1940 2075
[2,] 1810 1970 2130 2290 2450
[3,] 2085 2270 2455 2640 2825
[4,] 2360 2570 2780 2990 3200
[5,] 2635 2870 3105 3340 3575
>
> A <- matrix (data = c(1,3,2,4,2,4,3,5,1,6,7,2,1,5,6,7), nrow = 4, byrow =
TRUE )
> A
     [,1] [,2] [,3] [,4]
[1,]    1    3    2    4
[2,]    2    4    3    5
[3,]    1    6    7    2
[4,]    1    5    6    7
> B <- matrix (data = c(1, 2, 3, 4), nrow = 4)
> B
     [,1]
[1,]    1
[2,]    2
[3,]    3
[4,]    4
> solve (a = A, b = B)
           [,1]
[1,]  0.6153846
[2,] -0.8461538
[3,]  1.0000000
[4,]  0.2307692
>
> I <- diag (x = 1, nrow = 5, ncol = 5)
> I
     [,1] [,2] [,3] [,4] [,5]
[1,]    1    0    0    0    0
[2,]    0    1    0    0    0
[3,]    0    0    1    0    0
[4,]    0    0    0    1    0
[5,]    0    0    0    0    1
> A <- matrix (data = 1:25 , nrow = 5)
> A %*% I
     [,1] [,2] [,3] [,4] [,5]
```

```
[1,]    1    6   11   16   21
[2,]    2    7   12   17   22
[3,]    3    8   13   18   23
[4,]    4    9   14   19   24
[5,]    5   10   15   20   25
> I %*% A
     [,1] [,2] [,3] [,4] [,5]
[1,]    1    6   11   16   21
[2,]    2    7   12   17   22
[3,]    3    8   13   18   23
[4,]    4    9   14   19   24
[5,]    5   10   15   20   25
>
> A <- matrix (data = c(1,2,3,1,2,3,4,5,6,2,3,4,5,6,7,8,9,1,2,3,4,5,6,7,3), nrow
= 5)
> A
     [,1] [,2] [,3] [,4] [,5]
[1,]    1    3    3    8    4
[2,]    2    4    4    9    5
[3,]    3    5    5    1    6
[4,]    1    6    6    2    7
[5,]    2    2    7    3    3
> library (MASS)
> ginv (A)
           [,1]       [,2]       [,3]       [,4]          [,5]
[1,] -0.3333333  0.3333333  0.3333333 -0.3333333  1.179612e-16
[2,] -4.0888889  3.6444444 -1.2222222  0.8666667 -2.000000e-01
[3,] -0.3555556  0.2444444 -0.2222222  0.1333333  2.000000e-01
[4,] -0.1111111  0.2222222 -0.1111111  0.0000000  2.602085e-18
[5,]  3.8888889 -3.4444444  1.2222222 -0.6666667 -2.664535e-15
> ginv (A) %*% A
              [,1]          [,2]          [,3]          [,4]          [,5]
[1,]  1.000000e+00 -1.540434e-15 -9.506285e-16 -5.342948e-16
-1.866562e-15
[2,]  8.881784e-16  1.000000e+00 -5.329071e-15 -1.287859e-14
-2.464695e-14
[3,] -5.551115e-17 -1.165734e-15  1.000000e+00 -8.881784e-16
-1.776357e-15
[4,] -2.168404e-16 -7.719519e-16 -7.589415e-16  1.000000e+00
-1.213439e-15
[5,]  0.000000e+00  1.953993e-14  6.217249e-15  1.265654e-14
1.000000e+00
> A %*% ginv (A)
              [,1]         [,2]          [,3]         [,4]          [,5]
[1,]  1.000000e+00 1.776357e-15 -1.776357e-15 2.220446e-15 -1.200429e-15
[2,] -7.105427e-15 1.000000e+00 -1.776357e-15 1.332268e-15 -5.316927e-16
[3,] -3.552714e-15 0.000000e+00  1.000000e+00 1.332268e-15  1.136244e-16
[4,]  0.000000e+00 3.552714e-15  0.000000e+00 1.000000e+00  2.272488e-16
[5,] -5.329071e-15 5.329071e-15 -8.881784e-16 1.776357e-15  1.000000e+00
>
> A <- matrix (data = c(1, 3, 2, 4, 2, 4, 3, 5, 1, 6, 7, 2, 1, 5, 6, 7), nrow =
4, byrow = TRUE )
> A
     [,1] [,2] [,3] [,4]
[1,]    1    3    2    4
[2,]    2    4    3    5
[3,]    1    6    7    2
[4,]    1    5    6    7
> B <- matrix (data = c(1, 2, 3, 4), nrow = 4)
> B
     [,1]
[1,]    1
[2,]    2
[3,]    3
[4,]    4
> library (MASS )
> X <- ginv (A) %*% B
> X
           [,1]
[1,]  0.6153846
[2,] -0.8461538
[3,]  1.0000000
[4,]  0.2307692
>
> A <- matrix (data = c(1,3,2,4,2,4,3,5,1,6,7,2,1,5,6,7), nrow = 4, byrow =
TRUE )
> A
     [,1] [,2] [,3] [,4]
[1,]    1    3    2    4
[2,]    2    4    3    5
[3,]    1    6    7    2
[4,]    1    5    6    7
> det (A)
[1] -39
>
> lpNorm <- function (A, p) {
+    if (p >= 1 & dim (A)[[ 2]] == 1 && is.infinite (p) == FALSE ) {
+       sum ((apply (X = A, MARGIN = 1, FUN = abs)) ** p) ** (1 / p)
+    } else if (p >= 1 & dim (A)[[ 2]] == 1 & is.infinite (p)) {
+       max (apply (X = A, MARGIN = 1, FUN = abs)) # Max Norm
+    } else {
+       invisible (NULL )
+    }
+ }
> lpNorm (A = matrix (data = 1:10 ), p = 1)
[1] 55
> lpNorm (A = matrix (data = 1:10 ), p = 2) # Euclidean Distance
[1] 19.62142
> lpNorm (A = matrix (data = 1:10 ), p = 3)
[1] 14.46245
> lpNorm (A = matrix (data = -100 :10 ), p = Inf )
[1] 100
>
> lpNorm (A = matrix (data = rep (0, 10 )) , p = 1) == 0
[1] TRUE
> lpNorm (A = matrix (data = 1:10 ) + matrix (data = 11 :20 ), p = 1) <=
lpNorm (A = matrix (data = 1:10 ), p = 1) + lpNorm (A = matrix (data = 11
:20 ), p = 1)
[1] TRUE
> tempFunc <- function (i) {
+   lpNorm (A = i * matrix (data = 1:10 ), p = 1) == abs (i) * lpNorm (A =
matrix (data = 1:10 ), p = 1)
+ }
> all (sapply (X = -10 :10 , FUN = tempFunc ))
[1] TRUE
>
> frobeniusNorm <- function (A) {
+   (sum (( as.numeric (A)) ** 2)) ** (1 / 2)
+ }
> frobeniusNorm (A = matrix (data = 1:25 , nrow = 5))
[1] 74.33034
>
> A <- diag (x = c(1:5, 6, 1, 2, 3, 4), nrow = 10 )
> A
      [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10]
 [1,]    1    0    0    0    0    0    0    0    0     0
 [2,]    0    2    0    0    0    0    0    0    0     0
 [3,]    0    0    3    0    0    0    0    0    0     0
 [4,]    0    0    0    4    0    0    0    0    0     0
 [5,]    0    0    0    0    5    0    0    0    0     0
 [6,]    0    0    0    0    0    6    0    0    0     0
 [7,]    0    0    0    0    0    0    1    0    0     0
 [8,]    0    0    0    0    0    0    0    2    0     0
 [9,]    0    0    0    0    0    0    0    0    3     0
[10,]    0    0    0    0    0    0    0    0    0     4
> X <- matrix (data = 21 :30 )
> X
      [,1]
 [1,]   21
 [2,]   22
 [3,]   23
 [4,]   24
 [5,]   25
 [6,]   26
 [7,]   27
 [8,]   28
 [9,]   29
[10,]   30
> A %*% X
      [,1]
 [1,]   21
 [2,]   44
 [3,]   69
 [4,]   96
 [5,]  125
 [6,]  156
 [7,]   27
 [8,]   56
 [9,]   87
[10,]  120
> library (MASS )
> ginv (A)
      [,1] [,2]      [,3] [,4] [,5]      [,6] [,7] [,8]      [,9] [,10]
 [1,]    1  0.0 0.0000000 0.00  0.0 0.0000000    0  0.0 0.0000000  0.00
 [2,]    0  0.5 0.0000000 0.00  0.0 0.0000000    0  0.0 0.0000000  0.00
 [3,]    0  0.0 0.3333333 0.00  0.0 0.0000000    0  0.0 0.0000000  0.00
 [4,]    0  0.0 0.0000000 0.25  0.0 0.0000000    0  0.0 0.0000000  0.00
 [5,]    0  0.0 0.0000000 0.00  0.2 0.0000000    0  0.0 0.0000000  0.00
 [6,]    0  0.0 0.0000000 0.00  0.0 0.1666667    0  0.0 0.0000000  0.00
 [7,]    0  0.0 0.0000000 0.00  0.0 0.0000000    1  0.0 0.0000000  0.00
 [8,]    0  0.0 0.0000000 0.00  0.0 0.0000000    0  0.5 0.0000000  0.00
 [9,]    0  0.0 0.0000000 0.00  0.0 0.0000000    0  0.0 0.3333333  0.00
[10,]    0  0.0 0.0000000 0.00  0.0 0.0000000    0  0.0 0.0000000  0.25
>
> A <- matrix (data = c(1, 2, 2, 1), nrow = 2)
> A
     [,1] [,2]
[1,]    1    2
[2,]    2    1
> all (A == t(A))
[1] TRUE
>
> lpNorm (A = matrix (data = c(1, 0, 0, 0)) , p = 2)
[1] 1
>
> X <- matrix (data = c(11 , 0, 0, 0))
> Y <- matrix (data = c(0, 11 , 0, 0))
> all (t(X) %*% Y == 0)
[1] TRUE
>
> X <- matrix (data = c(1, 0, 0, 0))
> Y <- matrix (data = c(0, 1, 0, 0))
> lpNorm (A = X, p = 2) == 1
[1] TRUE
> lpNorm (A = Y, p = 2) == 1
[1] TRUE
> all (t(X) %*% Y == 0)
[1] TRUE
>
> A <- matrix (data = c(1, 0, 0, 0, 1, 0, 0, 0, 1), nrow = 3, byrow = TRUE
)
> A
     [,1] [,2] [,3]
[1,]    1    0    0
[2,]    0    1    0
[3,]    0    0    1
> all (t(A) %*% A == A %*% t(A))
[1] TRUE
> all (t(A) %*% A == diag (x = 1, nrow = 3))
[1] TRUE
> library (MASS )
> all (t(A) == ginv (A))
```

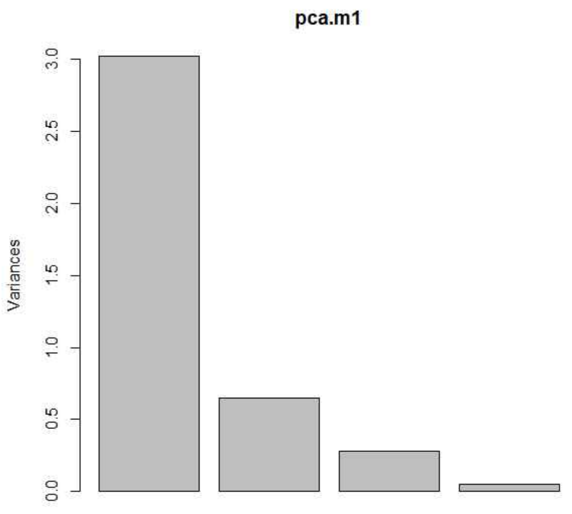
```
[1] TRUE
>
> A <- matrix (data = 1:25 , nrow = 5, byrow = TRUE )
> A
     [,1] [,2] [,3] [,4] [,5]
[1,]    1    2    3    4    5
[2,]    6    7    8    9   10
[3,]   11   12   13   14   15
[4,]   16   17   18   19   20
[5,]   21   22   23   24   25
> y <- eigen (x = A)
> library (MASS )
> all.equal (y$vectors %*% diag (y$values ) %*% ginv (y$vectors ), A)
[1] "Modes: complex, numeric"
>
> A <- matrix (data = 1:36 , nrow = 6, byrow = TRUE )
> A
     [,1] [,2] [,3] [,4] [,5] [,6]
[1,]    1    2    3    4    5    6
[2,]    7    8    9   10   11   12
[3,]   13   14   15   16   17   18
[4,]   19   20   21   22   23   24
[5,]   25   26   27   28   29   30
[6,]   31   32   33   34   35   36
> y <- svd (x = A)
> y
$`d`
[1] 1.272064e+02 4.952580e+00 8.605504e-15 3.966874e-15 1.157863e-15
[6] 2.985345e-16

$u
            [,1]        [,2]        [,3]         [,4]       [,5]
[1,] -0.06954892 -0.72039744  0.66413087 -0.157822256  0.0891868
[2,] -0.18479698 -0.51096788 -0.63156759 -0.090533384  0.4418991
[3,] -0.30004504 -0.30153832 -0.34646216 -0.006631248 -0.4698923
[4,] -0.41529310 -0.09210875  0.04190338  0.247123228 -0.6149437
[5,] -0.53054116  0.11732081  0.16119572  0.676892104  0.4260339
[6,] -0.64578922  0.32675037  0.11079978 -0.669028444  0.1277162
             [,6]
[1,] -0.047523836
[2,]  0.320020543
[3,] -0.691497505
[4,]  0.614870789
[5,] -0.197712056
[6,]  0.001842064

$v
           [,1]        [,2]       [,3]        [,4]        [,5]       [,6]
[1,] -0.3650545  0.62493577  0.5994575  0.02705792  0.21514480  0.2642381
[2,] -0.3819249  0.38648609 -0.4791520 -0.46790018  0.19629562 -0.4665969
[3,] -0.3987952  0.14803642 -0.1071927  0.67748875 -0.49153635 -0.3270449
[4,] -0.4156655 -0.09041326 -0.2351992 -0.33258158 -0.51273132  0.6246800
[5,] -0.4325358 -0.32886294 -0.2887029  0.36900805  0.63916519  0.2769719
[6,] -0.4494062 -0.56731262  0.5107893 -0.27307296 -0.04633795 -0.3722482

> all.equal (y$u %*% diag (y$d) %*% t(y$v), A)
[1] TRUE
>
> A <- matrix (data = 1:25 , nrow = 5)
> A
     [,1] [,2] [,3] [,4] [,5]
[1,]    1    6   11   16   21
[2,]    2    7   12   17   22
[3,]    3    8   13   18   23
[4,]    4    9   14   19   24
[5,]    5   10   15   20   25
> B <- ginv (A)
> B
       [,1]  [,2]          [,3]   [,4]   [,5]
[1,] -0.152 -0.08 -8.000000e-03  0.064  0.136
[2,] -0.096 -0.05 -4.000000e-03  0.042  0.088
[3,] -0.040 -0.02  1.170938e-17  0.020  0.040
[4,]  0.016  0.01  4.000000e-03 -0.002 -0.008
[5,]  0.072  0.04  8.000000e-03 -0.024 -0.056
> y <- svd (A)
> all.equal (y$v %*% ginv (diag (y$d)) %*% t(y$u), B)
[1] TRUE
>
> A <- diag (x = 1:10 )
> A
      [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10]
 [1,]    1    0    0    0    0    0    0    0    0     0
 [2,]    0    2    0    0    0    0    0    0    0     0
 [3,]    0    0    3    0    0    0    0    0    0     0
 [4,]    0    0    0    4    0    0    0    0    0     0
 [5,]    0    0    0    0    5    0    0    0    0     0
 [6,]    0    0    0    0    0    6    0    0    0     0
 [7,]    0    0    0    0    0    0    7    0    0     0
 [8,]    0    0    0    0    0    0    0    8    0     0
 [9,]    0    0    0    0    0    0    0    0    9     0
[10,]    0    0    0    0    0    0    0    0    0    10
> library (psych )
> tr (A)
[1] 55
> alternativeFrobeniusNorm <- function (A) {
+   sqrt (tr (t(A) %*% A))
+ }
> alternativeFrobeniusNorm (A)
[1] 19.62142
> frobeniusNorm (A)
[1] 19.62142
> all.equal (tr (A), tr (t(A)))
[1] TRUE
> A <- diag (x = 1:5)
> A
     [,1] [,2] [,3] [,4] [,5]
```

```
[1,]    1    0    0    0    0
[2,]    0    2    0    0    0
[3,]    0    0    3    0    0
[4,]    0    0    0    4    0
[5,]    0    0    0    0    5
> B <- diag (x = 6:10 )
> B
     [,1] [,2] [,3] [,4] [,5]
[1,]    6    0    0    0    0
[2,]    0    7    0    0    0
[3,]    0    0    8    0    0
[4,]    0    0    0    9    0
[5,]    0    0    0    0   10
> C <- diag (x = 11 :15 )
> C
     [,1] [,2] [,3] [,4] [,5]
[1,]   11    0    0    0    0
[2,]    0   12    0    0    0
[3,]    0    0   13    0    0
[4,]    0    0    0   14    0
[5,]    0    0    0    0   15
> all.equal (tr (A %*% B %*% C), tr (C %*% A %*% B))
[1] TRUE
> all.equal (tr (C %*% A %*% B), tr (B %*% C %*% A))
[1] TRUE
```

< 2. Singular Value Decomposition >

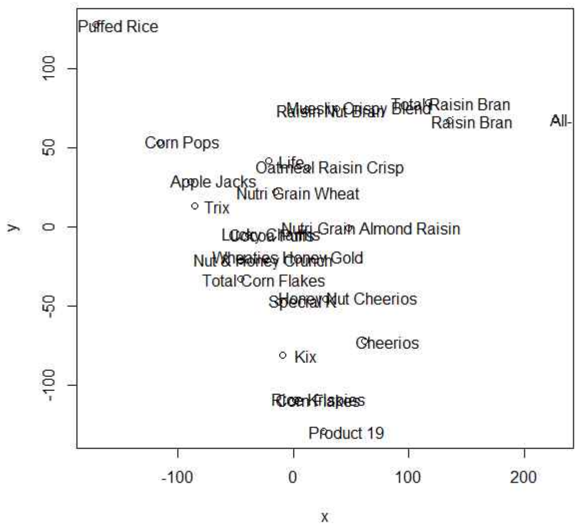
```
> education.by.readership <-
matrix(c(5,18,19,12,3,7,46,29,40,7,2,20,39,49,16),
+   nrow <- 5)
> dimnames(education.by.readership) <- list(
+      "Level of education" <- c("Some primary", "Primary completed",
"Some secondary", "Secondary completed", "sum tertiary"),
+      "Category of readership" <- c("Glance", "Fairly thorough", "Very
thorough"))
> print(education.by.readership)
                    Glance Fairly thorough Very thorough
Some primary             5               7             2
Primary completed       18              46            20
Some secondary          19              29            39
Secondary completed     12              40            49
sum tertiary             3               7            16
> O <- education.by.readership / sum(education.by.readership)
> print(O)
                         Glance Fairly thorough Very thorough
Some primary        0.016025641      0.02243590   0.006410256
Primary completed   0.057692308      0.14743590   0.064102564
Some secondary      0.060897436      0.09294872   0.125000000
Secondary completed 0.038461538      0.12820513   0.157051282
sum tertiary        0.009615385      0.02243590   0.051282051
> rowSums(O)
       Some primary   Primary completed      Some secondary
         0.04487179          0.26923077          0.27884615
Secondary completed        sum tertiary
         0.32371795          0.08333333
> colSums(O)
         Glance Fairly thorough   Very thorough
      0.1826923       0.4134615       0.4038462
> E <- rowSums(O) %o% colSums(O)
> print(E)
                         Glance Fairly thorough Very thorough
Some primary        0.008197732      0.01855276    0.01812130
Primary completed   0.049186391      0.11131657    0.10872781
Some secondary      0.050943047      0.11529216    0.11261095
Secondary completed 0.059140779      0.13384492    0.13073225
sum tertiary        0.015224359      0.03445513    0.03365385
> Z <- (O - E) / sqrt(E)
> print(Z)
                         Glance Fairly thorough Very thorough
Some primary         0.08645676      0.02850876   -0.08699634
Primary completed    0.03835294      0.10825794   -0.13533506
Some secondary       0.04410341     -0.06580368    0.03691882
Secondary completed -0.08503370     -0.01541566    0.07279115
sum tertiary        -0.04545838     -0.06475149    0.09609278
>
> SVD = svd(Z)
> print(SVD)
$`d`
[1] 2.652708e-01 1.135421e-01 2.718254e-17

$u
           [,1]        [,2]       [,3]
[1,] -0.4386666 -0.42375592 -0.3714480
[2,] -0.6516462  0.35501142 -0.4906752
[3,]  0.1603076 -0.67246939 -0.2423522
[4,]  0.3711005  0.48847409 -0.3785281
[5,]  0.4685240 -0.05979793 -0.6474922

$v
           [,1]        [,2]       [,3]
[1,] -0.4097795 -0.80584644 -0.4274252
[2,] -0.4887795  0.58960413 -0.6430097
[3,]  0.7701788 -0.05457549 -0.6354889

>
> sum(SVD$d * SVD$u[5, ] * SVD$v[2, ])
[1] -0.06475149
>
> SVD$u %*% diag(SVD$d) %*% t(SVD$v)
            [,1]        [,2]        [,3]
[1,]  0.08645676  0.02850876 -0.08699634
[2,]  0.03835294  0.10825794 -0.13533506
[3,]  0.04410341 -0.06580368  0.03691882
```

```
[4,] -0.08503370 -0.01541566  0.07279115
[5,] -0.04545838 -0.06475149  0.09609278
>
> variance.explained = prop.table(svd(Z)$d^2)
>
> library(MASS)
> a <- matrix(c(1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0,
+   0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1), 9, 4)
> print(a)
      [,1] [,2] [,3] [,4]
 [1,]    1    1    0    0
 [2,]    1    1    0    0
 [3,]    1    1    0    0
 [4,]    1    0    1    0
 [5,]    1    0    1    0
 [6,]    1    0    1    0
 [7,]    1    0    0    1
 [8,]    1    0    0    1
 [9,]    1    0    0    1
> a.svd <- svd(a)
> a.svd$d
[1] 3.464102e+00 1.732051e+00 1.732051e+00 1.922963e-16
> ds <- diag(1/a.svd$d[1:3])
> u <- a.svd$u
> v <- a.svd$v
> us <- as.matrix(u[, 1:3])
> vs <- as.matrix(v[, 1:3])
> (a.ginv <- vs %*% ds %*% t(us))
            [,1]        [,2]        [,3]        [,4]        [,5]        [,6]
[1,]  0.08333333  0.08333333  0.08333333  0.08333333  0.08333333
0.08333333
[2,]  0.25000000  0.25000000  0.25000000 -0.08333333 -0.08333333
-0.08333333
[3,] -0.08333333 -0.08333333 -0.08333333  0.25000000  0.25000000
0.25000000
[4,] -0.08333333 -0.08333333 -0.08333333 -0.08333333 -0.08333333
-0.08333333
            [,7]        [,8]        [,9]
[1,]  0.08333333  0.08333333  0.08333333
[2,] -0.08333333 -0.08333333 -0.08333333
[3,] -0.08333333 -0.08333333 -0.08333333
[4,]  0.25000000  0.25000000  0.25000000
> # using the function ginv defined in MASS
> ginv(a)
            [,1]        [,2]        [,3]        [,4]        [,5]        [,6]
[1,]  0.08333333  0.08333333  0.08333333  0.08333333  0.08333333
0.08333333
[2,]  0.25000000  0.25000000  0.25000000 -0.08333333 -0.08333333
-0.08333333
[3,] -0.08333333 -0.08333333 -0.08333333  0.25000000  0.25000000
0.25000000
[4,] -0.08333333 -0.08333333 -0.08333333 -0.08333333 -0.08333333
-0.08333333
            [,7]        [,8]        [,9]
[1,]  0.08333333  0.08333333  0.08333333
[2,] -0.08333333 -0.08333333 -0.08333333
[3,] -0.08333333 -0.08333333 -0.08333333
[4,]  0.25000000  0.25000000  0.25000000
>
> #버전이 바뀌면서 ReadImages가 OpenImageR로 바뀜
> #install.packages("OpenImageR")
> library(OpenImageR)
> x <- readImage("pansy.jpg")
> dim(x)
[1]  648 1152    3
> #버전이 바뀌면서 함수도 모두 바뀜, 실행 불가능
plot(x, useRaster = TRUE)
r <- imagematrix(x, type = "grey")
plot(r, useRaster = TRUE)
r.svd <- svd(r)
d <- diag(r.svd$d)
dim(d)
u <- r.svd$u
v <- r.svd$v
plot(1:length(r.svd$d), r.svd$d)
# first approximation
u1 <- as.matrix(u[-1, 1])
v1 <- as.matrix(v[-1, 1])
d1 <- as.matrix(d[1, 1])
l1 <- u1 %*% d1 %*% t(v1)
l1g <- imagematrix(l1, type = "grey")
plot(l1g, useRaster = TRUE)
# more approximation
depth <- 5
us <- as.matrix(u[, 1:depth])
vs <- as.matrix(v[, 1:depth])
ds <- as.matrix(d[1:depth, 1:depth])
ls <- us %*% ds %*% t(vs)
lsg <- imagematrix(ls, type = "grey")
plot(lsg, useRaster = TRUE)
>
> library(foreign)
> auto <- read.dta("http://statistics.ats.ucla.edu/stat/data/auto.dta")
> pca.m1 <- prcomp(~trunk + weight + length + headroom, data = auto,
scale = TRUE)
> screeplot(pca.m1)
```

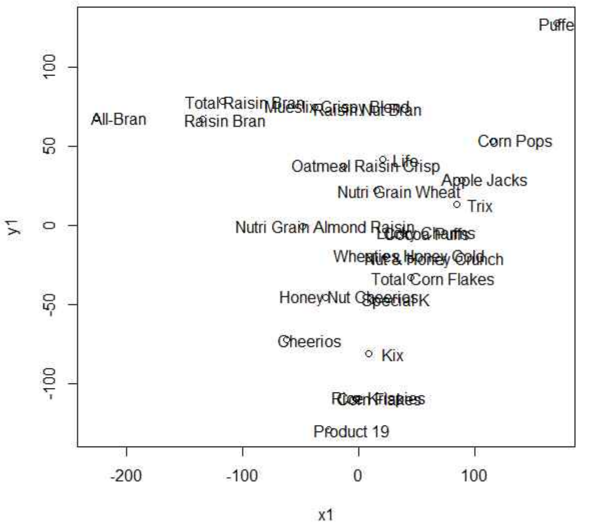


```
> # spectral decomposition: eigen values and eigen vectors
> xvars <- with(auto, cbind(trunk, weight, length, headroom))
> corr <- cor(xvars)
> a <- eigen(corr)
> (std <- sqrt(a$values))
[1] 1.7378931 0.8074981 0.5264150 0.2248592
> (rotation <- a$vectors)
           [,1]       [,2]       [,3]         [,4]
[1,] -0.5067777 -0.2326998  0.8249462  0.092145980
[2,] -0.5220823  0.4535800 -0.2677106  0.670839942
[3,] -0.5361131  0.3903201 -0.1370497 -0.735833101
[4,] -0.4280061 -0.7666591 -0.4785521 -0.005704251
> # svd approach
> df <- nrow(xvars) - 1
> zvars <- scale(xvars)
> z.svd <- svd(zvars)
> z.svd$d/sqrt(df)
[1] 1.7378931 0.8074981 0.5264150 0.2248592
> z.svd$v
          [,1]       [,2]       [,3]         [,4]
[1,] 0.5067777 -0.2326998  0.8249462 -0.092145980
[2,] 0.5220823  0.4535800 -0.2677106 -0.670839942
[3,] 0.5361131  0.3903201 -0.1370497  0.735833101
[4,] 0.4280061 -0.7666591 -0.4785521  0.005704251
>
> cnut <- read.dta("http://statistics.ats.ucla.edu/stat/data/cerealnut.dta")
> # centering the variables
> mds.data <- as.matrix(sweep(cnut[, -1], 2, colMeans(cnut[, -1])))
> dismat <- dist(mds.data)
> mds.m1 <- cmdscale(dismat, k = 8, eig = TRUE)
```

```
> mds.m1$eig
 [1]  1.584379e+05  1.087288e+05  1.056264e+04  3.826785e+02
6.976171e+01
 [6]  1.252082e+01  5.755998e+00  2.224324e+00  4.513969e-12
4.508111e-12
[11]  4.121611e-12  3.188527e-12  3.150030e-12  2.297497e-12
2.091059e-12
[16]  1.246190e-12  1.131813e-12  8.794901e-13  2.967892e-13
-1.382636e-12
[21] -1.452732e-12 -1.574794e-12 -1.876268e-12 -5.916330e-12
-2.520931e-11
> mds.m1 <- cmdscale(dismat, k = 2, eig = TRUE)
> x <- mds.m1$points[, 1]
> y <- mds.m1$points[, 2]
> plot(x, y)
> text(x + 20, y, label = cnut$brand)
```

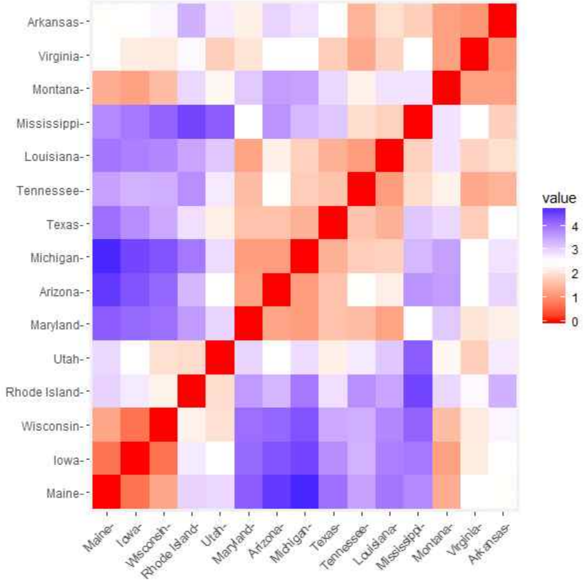


```
> # eigenvalues
> xx <- svd(mds.data %*% t(mds.data))
> xx$d
 [1] 1.584379e+05 1.087288e+05 1.056264e+04 3.826785e+02 6.976171e+01
 [6] 1.252082e+01 5.755998e+00 2.224324e+00 1.576321e-11 9.903742e-12
[11] 7.190968e-12 4.712199e-12 4.152571e-12 3.030837e-12 2.767589e-12
[16] 2.082324e-12 1.971417e-12 1.496531e-12 1.258080e-12 1.045736e-12
[21] 7.934340e-13 7.346559e-13 2.088189e-13 1.653877e-13 8.383459e-14
> # coordinates
> xxd <- xx$v %*% sqrt(diag(xx$d))
> x1 <- xxd[, 1]
> y1 <- xxd[, 2]
> plot(x1, y1)
> text(x1 + 20, y1, label = cnut$brand)
```



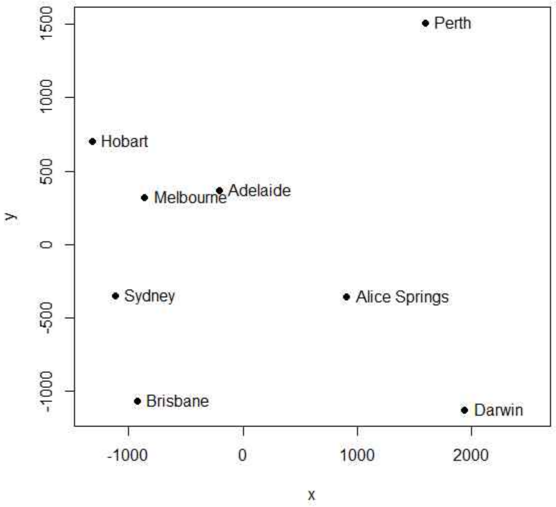
< 3. Similarity and Dissimilarity >

```
> # Subset of the data
> set.seed (123 )
> ss <- sample (1:50 , 15 ) # Take 15 random rows
> df <- USArrests [ss , ] # Subset the 15 rows
> df.scaled <- scale (df ) # Standardize the
>
> dist.eucl <- dist (df.scaled , method = "euclidean" )
>
> # Reformat as a matrix
> # Subset the first 3 columns and rows and Round the values
> round (as.matrix (dist.eucl )[ 1:3, 1:3], 1)
             Iowa Rhode Island Maryland
Iowa          0.0          2.8      4.1
Rhode Island  2.8          0.0      3.6
Maryland      4.1          3.6      0.0
>
> # Compute
> #install.packages("factoextra")
> library ("factoextra" )
> dist.cor <- get_dist (df.scaled , method = "pearson" )
> # Display a subset
> round (as.matrix (dist.cor )[ 1:3, 1:3], 1)
             Iowa Rhode Island Maryland
Iowa          0.0          0.4      1.9
Rhode Island  0.4          0.0      1.5
Maryland      1.9          1.5      0.0
>
> library (cluster )
> # Load data
> data (flower )
> head (flower , 3)
  V1 V2 V3 V4 V5 V6  V7 V8
1  0  1  1  4  3 15  25 15
2  1  0  0  2  1  3 150 50
3  0  1  0  3  3  1 150 50
> # Data structure
> str (flower )
'data.frame':   18 obs. of  8 variables:
 $ V1: Factor w/ 2 levels "0","1": 1 2 1 1 1 1 1 1 2 2 ...
 $ V2: Factor w/ 2 levels "0","1": 2 1 2 1 2 2 1 1 2 2 ...
 $ V3: Factor w/ 2 levels "0","1": 2 1 1 2 1 1 1 2 1 1 ...
 $ V4: Factor w/ 5 levels "1","2","3","4",..: 4 2 3 4 5 4 4 2 3 5 ...
 $ V5: Ord.factor w/ 3 levels "1"<"2"<"3": 3 1 3 2 2 3 3 2 1 2 ...
 $ V6: Ord.factor w/ 18 levels "1"<"2"<"3"<"4"<..: 15 3 1 16 2 12 13 7 4 14
...
 $ V7: num  25 150 150 125 20 50 40 100 25 100 ...
 $ V8: num  15 50 50 50 15 40 20 15 15 60 ...
> dd <- daisy (flower )
> round (as.matrix (dd )[ 1:3, 1:3], 2)
     1    2    3
1 0.00 0.89 0.53
2 0.89 0.00 0.51
3 0.53 0.51 0.00
>
> library (factoextra )
> fviz_dist (dist.eucl )
```

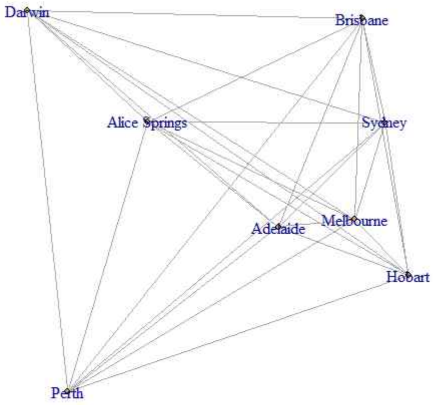


< 4. Multidimensional Scaling >

```
> url <- "http://rosetta.reltech.org/TC/v1 5/Mapping/data/dist -Aus.csv"
> #dist.au <- read.csv(url)
>
> dist.au <- read.csv("dist-Aus.csv")
> dist.au
   X    A   AS    B    D    H    M    P    S
1  A    0 1328 1600 2616 1161  653 2130 1161
2 AS 1328    0 1962 1289 2463 1889 1991 2026
3  B 1600 1962    0 2846 1788 1374 3604  732
4  D 2616 1289 2846    0 3734 3146 2652 3146
5  H 1161 2463 1788 3734    0  598 3008 1057
6  M  653 1889 1374 3146  598    0 2720  713
7  P 2130 1991 3604 2652 3008 2720    0 3288
8  S 1161 2026  732 3146 1057  713 3288    0
> row.names(dist.au) <- dist.au[, 1]
> dist.au <- dist.au[, -1]
> dist.au
      A   AS    B    D    H    M    P    S
A     0 1328 1600 2616 1161  653 2130 1161
AS 1328    0 1962 1289 2463 1889 1991 2026
B  1600 1962    0 2846 1788 1374 3604  732
D  2616 1289 2846    0 3734 3146 2652 3146
H  1161 2463 1788 3734    0  598 3008 1057
M   653 1889 1374 3146  598    0 2720  713
P  2130 1991 3604 2652 3008 2720    0 3288
S  1161 2026  732 3146 1057  713 3288    0
>
> fit <- cmdscale(dist.au, eig = TRUE, k = 2)
> x <- fit$points[, 1]
> y <- fit$points[, 2]
>
> plot(x, y, pch = 19, xlim = range(x) + c(0, 600))
> city.names <- c("Adelaide", "Alice Springs", "Brisbane", "Darwin", "Hobart",
"Melbourne", "Perth", "Sydney")
> text(x, y, pos = 4, labels = city.names)
```

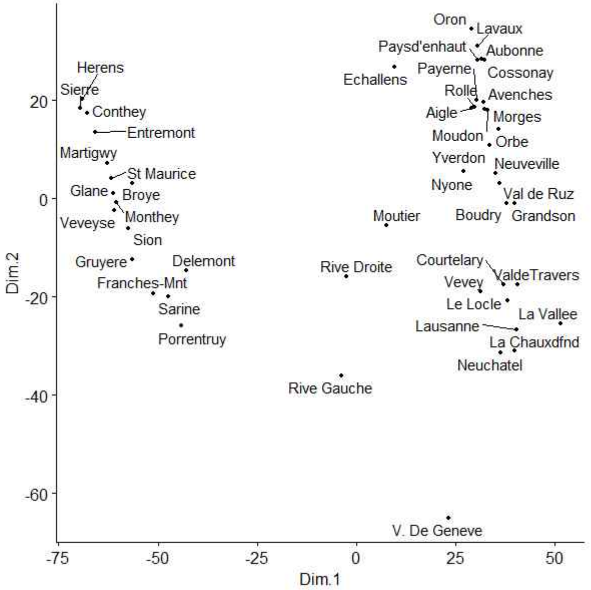


```
> #install.packages("igraph")
> library(igraph)
> g <- graph.full(nrow(dist.au))
> V(g)$label <- city.names
> layout <- layout.mds(g, dist = as.matrix(dist.au))
> plot(g, layout = layout, vertex.size = 3)
```

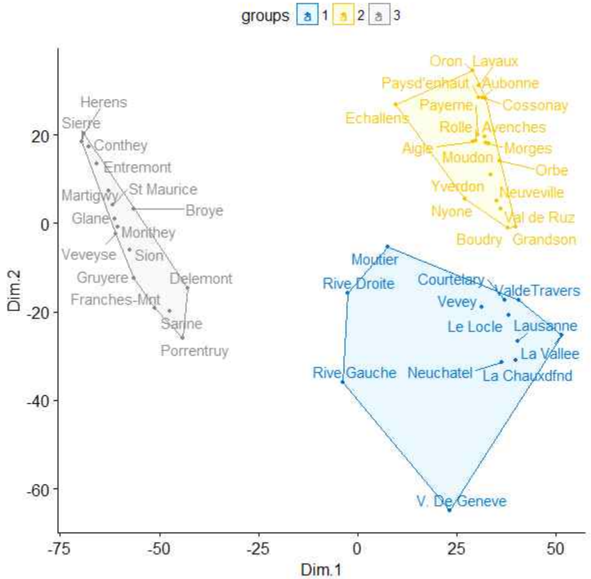


```
> data ("swiss" )
> head (swiss )
             Fertility Agriculture Examination Education Catholic
```

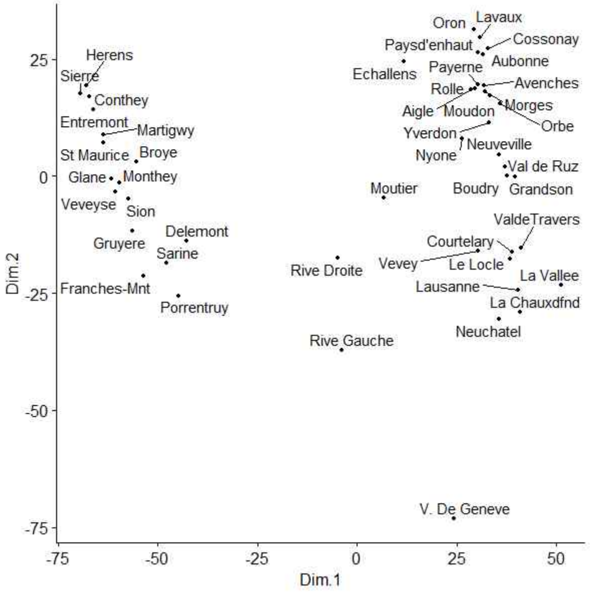
```
Courtelary         80.2          17.0              15           12      9.96
Delemont            83.1          45.1              6            9     84.84
Franches-Mnt        92.5           39.7              5            5     93.40
Moutier            85.8          36.5             12            7     33.77
Neuveville         76.9          43.5             17           15      5.16
Porrentruy         76.1          35.3              9            7     90.57
               Infant.Mortality
Courtelary                  22.2
Delemont                     22.2
Franches-Mnt                  20.2
Moutier                     20.3
Neuveville                  20.6
Porrentruy                   26.6
>
> # Load required packages
> library (magrittr)
> library (dplyr)
> library (ggpubr)
> # C ompute MDS
> ## The infix operator “%>% ” is not part of base R,
> ## but defined by the package magrittr .
> ## It works like a pipe , pass ing the LHS to first argument of RHS .
> mds <- swiss %>% dist () %>% cmdscale () %>% as_tibble ()
> colnames (mds ) <- c("Dim.1" , "Dim.2" )
> # Plot MDS
> ggscatter (mds , x = "Dim.1" , y = "Dim.2" ,
+   label = rownames (swiss ),
+   size = 1,
+   repel = TRUE )
```



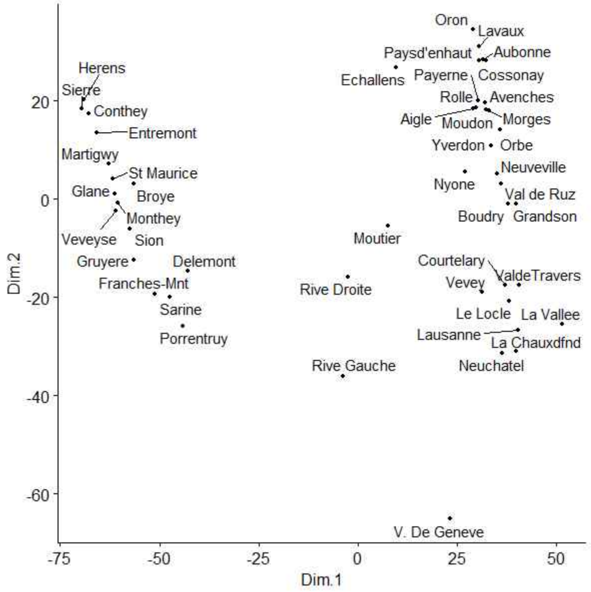
```
> # K -means clustering
> clust <- kmeans (mds , 3)$cluster %>% as.factor ()
> mds <- mds %>% mutate (groups = clust )
> # Plot and color by groups
> ggscatter (mds , x = "Dim.1" , y = "Dim.2" ,
+   label = rownames (swiss ),
+   color = "groups" ,
+   palette = "jco" ,
+   size = 1,
+   ellipse = TRUE ,
+   ellipse.type = "convex" ,
+   repel = TRUE )
```



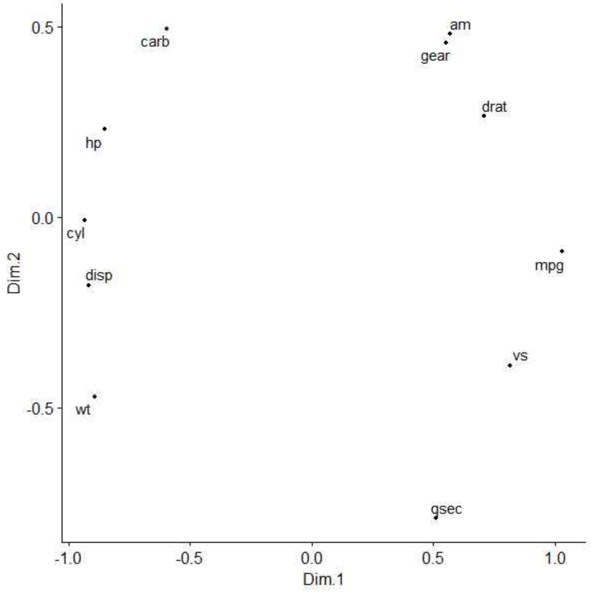
```
> library (magrittr )
> library (dplyr )
> library (ggpubr )
>
> # C ompute MDS
> library (MASS )
> mds <- swiss %>% dist () %>% isoMDS () %>% .$points %>% as_tibble ()
initial  value 5.463800
iter    5 value 4.499103
iter    5 value 4.495335
iter    5 value 4.492669
final  value 4.492669
converged
> colnames (mds ) <- c("Dim.1" , "Dim.2" )
> # Plot MDS
> ggscatter (mds , x = "Dim.1" , y = "Dim.2" ,
+   label = rownames (swiss ),
+   size = 1,
+   repel = TRUE )
```



```
> # Compute MDS
> library (MASS )
> mds <- swiss %>% dist () %>% sammon () %>% .$points %>% as_tibble ()
Initial stress        : 0.01959
stress after   0 iters: 0.01959
> colnames (mds ) <- c("Dim.1" , "Dim.2" )
> # Plot MDS
> ggscatter (mds , x = "Dim.1" , y = "Dim.2" ,
+   label = rownames (swiss ),
+   size = 1,
+   repel = TRUE )
```

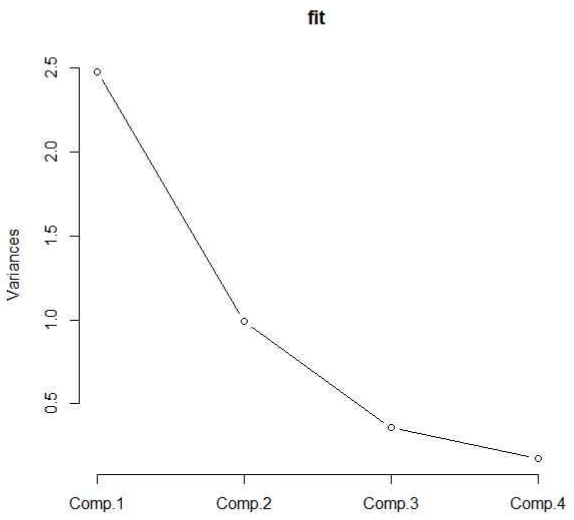


```
> res.cor <- cor (mtcars , method = "spearman" )
> mds.cor <- (1 - res.cor ) %>% cmdscale () %>% as_tibble ()
> colnames (mds.cor ) <- c("Dim.1" , "Dim.2" )
> ggscatter (mds.cor , x = "Dim.1" , y = "Dim.2" ,
+   size = 1,
+   label = colnames (res.cor ),
+   repel = TRUE )
```

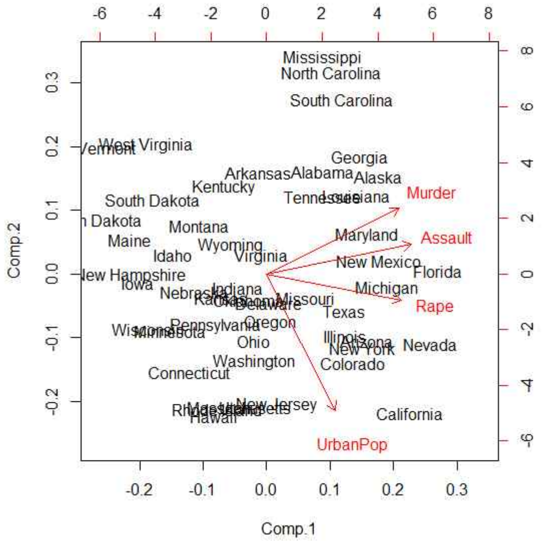


< 5. Principal Components Analysis >

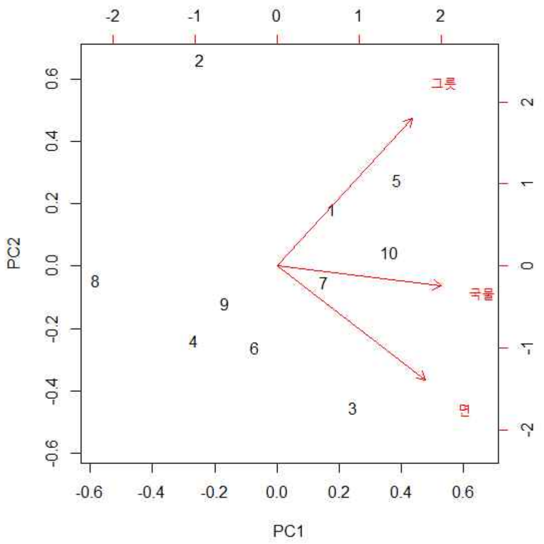
```
> library(datasets)
> data(USArrests)
> summary(USArrests)
     Murder          Assault         UrbanPop          Rape      
 Min.   : 0.800   Min.   : 45.0   Min.   :32.00   Min.   : 7.30  
 1st Qu.: 4.075   1st Qu.:109.0   1st Qu.:54.50   1st Qu.:15.07  
 Median : 7.250   Median :159.0   Median :66.00   Median :20.10  
 Mean   : 7.788   Mean   :170.8   Mean   :65.54   Mean   :21.23  
 3rd Qu.:11.250   3rd Qu.:249.0   3rd Qu.:77.75   3rd Qu.:26.18  
 Max.   :17.400   Max.   :337.0   Max.   :91.00   Max.   :46.00  
> myData <- USArrests
> fit <- princomp(myData, cor=TRUE)
> summary(fit)
Importance of components:
                          Comp.1    Comp.2    Comp.3     Comp.4
Standard deviation     1.5748783 0.9948694 0.5971291 0.41644938
Proportion of Variance 0.6200604 0.2474413 0.0891408 0.04335752
Cumulative Proportion  0.6200604 0.8675017 0.9566425 1.00000000
> loadings(fit)

Loadings:
         Comp.1 Comp.2 Comp.3 Comp.4
Murder    0.536  0.418  0.341  0.649
Assault   0.583  0.188  0.268 -0.743
UrbanPop  0.278 -0.873  0.378  0.134
Rape      0.543 -0.167 -0.818       

               Comp.1 Comp.2 Comp.3 Comp.4
SS loadings      1.00   1.00   1.00   1.00
Proportion Var   0.25   0.25   0.25   0.25
Cumulative Var   0.25   0.50   0.75   1.00
> plot(fit, type="lines")
```



```
> fit$scores
                    Comp.1      Comp.2      Comp.3       Comp.4
Alabama         0.98556588  1.13339238  0.44426879  0.156267145
Alaska          1.95013775  1.07321326 -2.04000333 -0.438583440
Arizona         1.76316354 -0.74595678 -0.05478082 -0.834652924
Arkansas       -0.14142029  1.11979678 -0.11457369 -0.182810896
California      2.52398013 -1.54293399 -0.59855680 -0.341996478
Colorado        1.51456286 -0.98755509 -1.09500699  0.001464887
Connecticut    -1.35864746 -1.08892789  0.64325757 -0.118469414
Delaware        0.04770931 -0.32535892  0.71863294 -0.881977637
Florida         3.01304227  0.03922851  0.57682949 -0.096284752
Georgia         1.63928304  1.27894240  0.34246008  1.076796812
Hawaii         -0.91265715 -1.57046001 -0.05078189  0.902806864
Idaho          -1.63979985  0.21097292 -0.25980134 -0.499104101
Illinois        1.37891072 -0.68184119  0.67749564 -0.122021292
Indiana        -0.50546136 -0.15156254 -0.22805484  0.424665700
Iowa           -2.25364607 -0.10405407 -0.16456432  0.017555916
Kansas         -0.79688112 -0.27016470 -0.02555331  0.206496428
Kentucky       -0.75085907  0.95844029  0.02836942  0.670556671
Louisiana       1.56481798  0.87105466  0.78348036  0.454728038
Maine          -2.39682949  0.37639158  0.06568239 -0.330459817
Maryland        1.76336939  0.42765519  0.15725013 -0.559069521
Massachusetts  -0.48616629 -1.47449650  0.60949748 -0.179598963
Michigan        2.10844115 -0.15539682 -0.38486858  0.102372019
Minnesota      -1.69268181 -0.63226125 -0.15307043  0.067316885
Mississippi     0.99649446  2.39379599  0.74080840  0.215508013
Missouri        0.69678733 -0.26335479 -0.37744383  0.225824461
Montana        -1.18545191  0.53687437 -0.24688932  0.123742227
Nebraska       -1.26563654 -0.19395373 -0.17557391  0.015892888
Nevada          2.87439454 -0.77560020 -1.16338049  0.314515476
New Hampshire  -2.38391541 -0.01808229 -0.03685539 -0.033137338
New Jersey        0.18156611 -1.44950571  0.76445355  0.243382700
New Mexico         1.98002375  0.14284878 -0.18369218 -0.339533597
New York           1.68257738 -0.82318414  0.64307509 -0.013484369
North Carolina   1.12337861  2.22800338  0.86357179 -0.954381667
North Dakota     -2.99222562  0.59911882 -0.30127728 -0.253987327
Ohio             -0.22596542 -0.74223824  0.03113912  0.473915911
Oklahoma          -0.31178286 -0.28785421  0.01530979  0.010332321
Oregon            0.05912208 -0.54141145 -0.93983298 -0.237780688
Pennsylvania     -0.88841582 -0.57110035  0.40062871  0.359061124
Rhode Island     -0.86377206 -1.49197842  1.36994570 -0.613569430
South Carolina   1.32072380  1.93340466  0.30053779 -0.131466685
South Dakota     -1.98777484  0.82334324 -0.38929333 -0.109571764
Tennessee         0.99974168  0.86025130 -0.18808295  0.652864291
Texas            1.35513821 -0.41248082  0.49206886  0.643195491
Utah             -0.55056526 -1.47150461 -0.29372804 -0.082314047
Vermont          -2.80141174  1.40228806 -0.84126309 -0.144889914
Virginia        -0.09633491  0.19973529 -0.01171254  0.211370813
Washington       -0.21690338 -0.97012418 -0.62487094 -0.220847793
West Virginia   -2.10858541  1.42484670 -0.10477467  0.131908831
Wisconsin        -2.07971417 -0.61126862  0.13886500  0.184103743
Wyoming           -0.62942666  0.32101297  0.24065923 -0.166651801
> biplot(fit)
```



```
> ramen <-matrix(c(2,1,5,2,3,4,4,1,3,5,4,5,3,2,5,3,4,2,3,5,5,1,4,3,5,2,3,1,2,3),
ncol=3)
> rownames(ramen) <- c(" 쇠고기라면 ", " 해물라면 ", " 얼큰라면 ", " 떡라면 ",
" 짬뽕라면 ",
+    "만두라면 ", " ", "치즈라면 ", " 된장라면 ", " 볶음라면 ", " 김치라면 ")
Error in dimnames(x) <- dn :
  'dimnames'의 길이 [1]가 배열의 크기와 같지 않습니다
> colnames(ramen) <- c(" 면", "그릇 ", " 국물 ")
> print( ramen )
        면 그릇   국물
 [1,]   2     4       5
 [2,]   1     5       1
 [3,]   5     3       4
 [4,]   2     2       3
 [5,]   3     5       5
 [6,]   4     3       2
 [7,]   4     4       3
 [8,]   1     2       1
 [9,]   3     3       2
[10,]   5     5       3
> pc<-prcomp( ramen , scale=TRUE)
> print( pc )
Standard deviations (1, .., p=3):
[1] 1.2541347 0.9022241 0.7830312

Rotation (n x k) = (3 x 3):
              PC1         PC2         PC3
 면     0.5715110 -0.6044710  0.5549685
그릇    0.5221161  0.7896069  0.3223595
 국물  0.6330639 -0.1055260 -0.7668731
> summary(pc)
Importance of components:
                          PC1    PC2    PC3
Standard deviation       1.2541 0.9022 0.7830
Proportion of Variance 0.5243 0.2713 0.2044
Cumulative Proportion  0.5243 0.7956 1.0000
>
> predict(pc)
              PC1         PC2           PC3
 [1,]  0.7119408  0.5216497 -1.373736133
 [2,] -0.9740499  1.8911205  0.645382316
 [3,]  0.9804158 -1.2947047 -0.002322692
 [4,] -1.0513965 -0.6781104 -0.864614382
 [5,]  1.5401350  0.7888582 -0.726820118
 [6,] -0.2766766 -0.7435735  0.683778524
 [7,]  0.6049920 -0.1436935  0.429217649
 [8,] -2.3084890 -0.1269792 -0.178513165
 [9,] -0.6600579 -0.3380821  0.311494336
[10,]  1.4331863  0.1235150  1.076133664
> biplot(pc)
```


-2
-1
0
1
2
0.6
0.4
0.2
0.0
-0.2
-0.4
-0.6
PC2
2
그릇
5
1
10
8
7
국물
9
4
6
3
면
-0.6
-0.4
-0.2
0.0
0.2
0.4
0.6
PC1