1. Intuition on SVD

1. (a)

ANS:

I find the rank using OBSERVATION with reasons, which are presented below. For the matrix with only 1 singular value, the rank-1 approximation is quite intuitive, since the rank-1 approximation is the original matrix themselves. But for those which have more-than-1 singular values, it is not so intuitive.

matrix	supposed (column) rank	Reason(s)
M1	1	all the vectors can be replicated by $(1,1,1,0,0)^T$
M2	1	all the vectors can be replicated by $(0,1,1,1,0)^T$
M3	1	all the vectors can be replicated by $(0,1,1,1,1)^T$
M4	2	all the vectors can be replicated by the combination of $(1,1,1,0,0)^T$ and $(0,0,0,1,1)^T$
M5	3	all the vectors can be replicated by the combination of $(1,1,1,0,0)^T$, $(1,1,1,1,1)^T$ and $(0,0,1,1,1)^T$
M6	2	all the vectors can be replicated by the combination of $(1,1,1,1,1)^T$ and $(1,1,0,1,1)^T$

1. (b)

```
the SVD decomposition of matrix M1
    diagonal valus of Σ
       [1] 3 0 0 0 0

    Matrix U
                 [,1]       [,2] [,3] [,4]       [,5]
       [1,] -0.5773503 -0.5773503    0    0 -0.5773503
       [2,] -0.5773503 -0.2113249    0    0  0.7886751
       [3,] -0.5773503  0.7886751    0    0 -0.2113249
       [4,]  0.0000000  0.0000000    1    0  0.0000000
       [5,]  0.0000000  0.0000000    0    1  0.0000000

    Matrix V
                 [,1]       [,2] [,3] [,4]       [,5]
       [1,] -0.5773503  0.0000000    0    0  0.8164966
       [2,] -0.5773503 -0.7071068    0    0 -0.4082483
       [3,] -0.5773503  0.7071068    0    0 -0.4082483
       [4,]  0.0000000  0.0000000    1    0  0.0000000
       [5,]  0.0000000  0.0000000    0    1  0.0000000

the SVD decomposition of matrix M2
    diagonal valus of Σ
        [1] 5.196152 0.000000 0.000000 0.000000 0.000000

    Matrix U
                 [,1]       [,2]       [,3] [,4] [,5]
       [1,]  0.0000000  0.0000000  0.0000000    0    1
       [2,] -0.5773503 -0.5773503 -0.5773503    0    0
       [3,] -0.5773503  0.7886751 -0.2113249    0    0
       [4,] -0.5773503 -0.2113249  0.7886751    0    0
```

```
     [5,]  0.0000000  0.0000000  0.0000000    1    0
```

Matrix V

```
              [,1]       [,2]       [,3] [,4] [,5]
     [1,]  0.0000000  0.0000000  0.0000000    0    1
     [2,] -0.6666667  0.7453560  0.0000000    0    0
     [3,] -0.3333333 -0.2981424 -0.8944272    0    0
     [4,] -0.6666667 -0.5962848  0.4472136    0    0
     [5,]  0.0000000  0.0000000  0.0000000    1    0
```

the SVD decomposition of matrix M3

diagonal valus of Σ

```
      [1] 3.464102e+00 3.140185e-16 0.000000e+00 0.000000e+00
```

Matrix U

```
          [,1]       [,2]          [,3] [,4]
     [1,]  0.0  0.0000000  0.000000e+00    1
     [2,] -0.5  0.8660254 -4.163336e-17    0
     [3,] -0.5 -0.2886751 -5.773503e-01    0
     [4,] -0.5 -0.2886751  7.886751e-01    0
     [5,] -0.5 -0.2886751 -2.113249e-01    0
```

Matrix V

```
              [,1]       [,2]       [,3] [,4]
     [1,]  0.0000000  0.0000000  0.0000000    1
     [2,] -0.5773503  0.8164966  0.0000000    0
     [3,] -0.5773503 -0.4082483 -0.7071068    0
     [4,] -0.5773503 -0.4082483  0.7071068    0
```

the SVD decomposition of matrix M4

diagonal valus of Σ

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

Matrix U

```
              [,1]       [,2]       [,3]       [,4]       [,5]
     [1,] -0.5773503  0.0000000  0.0000000 -0.5773503 -0.5773503
     [2,] -0.5773503  0.0000000  0.0000000 -0.2113249  0.7886751
     [3,] -0.5773503  0.0000000  0.0000000  0.7886751 -0.2113249
     [4,]  0.0000000 -0.7071068 -0.7071068  0.0000000  0.0000000
     [5,]  0.0000000 -0.7071068  0.7071068  0.0000000  0.0000000
```

Matrix V

```
              [,1]       [,2]       [,3]       [,4]       [,5]
     [1,] -0.5773503  0.0000000  0.0000000  0.0000000  0.8164966
     [2,] -0.5773503  0.0000000  0.0000000 -0.7071068 -0.4082483
     [3,] -0.5773503  0.0000000  0.0000000  0.7071068 -0.4082483
     [4,]  0.0000000 -0.7071068 -0.7071068  0.0000000  0.0000000
     [5,]  0.0000000 -0.7071068  0.7071068  0.0000000  0.0000000
```

the SVD decomposition of matrix M5

diagonal valus of Σ

```
      [1] 3.561553e+00 2.000000e+00 5.615528e-01 3.025104e-17 0.000000e+00
```

Matrix U

```
              [,1]          [,2]       [,3]          [,4]          [,5]
     [1,] -0.3941027 -5.000000e-01  0.3077061  7.071068e-01  8.417630e-17
     [2,] -0.3941027 -5.000000e-01  0.3077061 -7.071068e-01 -8.667745e-17
     [3,] -0.6154122 -1.665335e-16 -0.7882054  0.000000e+00  2.501145e-18
     [4,] -0.3941027  5.000000e-01  0.3077061 -1.110223e-16 -7.071068e-01
     [5,] -0.3941027  5.000000e-01  0.3077061  0.000000e+00  7.071068e-01
```

Matrix V

```
              [,1]          [,2]       [,3]          [,4]          [,5]
     [1,] -0.3941027 -5.000000e-01 -0.3077061 -7.071068e-01  0.000000e+00
     [2,] -0.3941027 -5.000000e-01 -0.3077061  7.071068e-01 -2.318729e-17
     [3,] -0.6154122 -1.665335e-16  0.7882054  2.220446e-16 -9.309503e-18
     [4,] -0.3941027  5.000000e-01 -0.3077061 -1.387779e-16 -7.071068e-01
```

```
    [5,] -0.3941027  5.000000e-01 -0.3077061 -8.326673e-17  7.071068e-01
```

the SVD decomposition of matrix M6

diagonal valus of Σ

```
     [1] 4.828427e+00 8.284271e-01 2.082505e-16 2.443061e-17 1.710569e-49
```

Matrix U

```
                [,1]       [,2]          [,3]          [,4]          [,5]
    [1,] -0.4619398 -0.1913417  8.586583e-01  1.127206e-01  0.000000e+00
    [2,] -0.4619398 -0.1913417 -3.924934e-01  7.719773e-01 -6.408810e-17
    [3,] -0.3826834  0.9238795 -1.110223e-16 -5.551115e-17  3.433180e-18
    [4,] -0.4619398 -0.1913417 -2.330825e-01 -4.423489e-01 -7.071068e-01
    [5,] -0.4619398 -0.1913417 -2.330825e-01 -4.423489e-01  7.071068e-01
```

Matrix V

```
                [,1]       [,2]          [,3]          [,4]          [,5]
    [1,] -0.4619398  0.1913417  8.552261e-01  1.363391e-01  0.000000e+00
    [2,] -0.4619398  0.1913417 -4.136171e-01  7.608685e-01 -4.230345e-17
    [3,] -0.3826834 -0.9238795  1.110223e-16 -8.326673e-17  1.576262e-17
    [4,] -0.4619398  0.1913417 -2.208045e-01 -4.486038e-01 -7.071068e-01
    [5,] -0.4619398  0.1913417 -2.208045e-01 -4.486038e-01  7.071068e-01
```

1. (c)

Marix	Original matrix	approximation using K=1
M1	[,1] [,2] [,3] [,4] [,5] [1,] 1 1 1 0 0 [2,] 1 1 1 0 0 [3,] 1 1 1 0 0 [4,] 0 0 0 0 0 [5,] 0 0 0 0 0	[,1] [,2] [,3] [,4] [,5] [1,] 1 1 1 0 0 [2,] 1 1 1 0 0 [3,] 1 1 1 0 0 [4,] 0 0 0 0 0 [5,] 0 0 0 0 0
M2	[,1] [,2] [,3] [,4] [,5] [1,] 0 0 0 0 0 [2,] 0 2 1 2 0 [3,] 0 2 1 2 0 [4,] 0 2 1 2 0 [5,] 0 0 0 0 0	[,1] [,2] [,3] [,4] [,5] [1,] 0 0 0 0 0 [2,] 0 2 1 2 0 [3,] 0 2 1 2 0 [4,] 0 2 1 2 0 [5,] 0 0 0 0 0
M3	[,1] [,2] [,3] [,4] [,5] [1,] 0 0 0 0 [2,] 0 1 1 1 [3,] 0 1 1 1 [4,] 0 1 1 1 [5,] 0 1 1 1	[,1] [,2] [,3] [,4] [,5] [1,] 0 0 0 0 [2,] 0 1 1 1 [3,] 0 1 1 1 [4,] 0 1 1 1 [5,] 0 1 1 1
M4	[,1] [,2] [,3] [,4] [,5] [1,] 1 1 1 0 0 [2,] 1 1 1 0 0 [3,] 1 1 1 0 0 [4,] 0 0 0 1 1 [5,] 0 0 0 1 1	[,1] [,2] [,3] [,4] [,5] [1,] 1 1 1 0 0 [2,] 1 1 1 0 0 [3,] 1 1 1 0 0 [4,] 0 0 0 0 0 [5,] 0 0 0 0 0
M5	[,1] [,2] [,3] [,4] [,5] [1,] 1 1 1 0 0 [2,] 1 1 1 0 0 [3,] 1 1 1 1 1 [4,] 0 0 1 1 1 [5,] 0 0 1 1 1	[,1] [,2] [,3] [,4] [,5] [1,] 0.553 0.553 0.864 0.553 0.553 [2,] 0.553 0.553 0.864 0.553 0.553 [3,] 0.864 0.864 1.349 0.864 0.864 [4,] 0.553 0.553 0.864 0.553 0.553 [5,] 0.553 0.553 0.864 0.553 0.553
M6	[,1] [,2] [,3] [,4] [,5] [1,] 1 1 1 1 1 [2,] 1 1 1 1 1 [3,] 1 1 0 1 1 [4,] 1 1 1 1 1 [5,] 1 1 1 1 1	[,1] [,2] [,3] [,4] [,5] [1,] 1.030 1.030 0.854 1.030 1.030 [2,] 1.030 1.030 0.854 1.030 1.030 [3,] 0.854 0.854 0.707 0.854 0.854 [4,] 1.030 1.030 0.854 1.030 1.030 [5,] 1.030 1.030 0.854 1.030 1.030

1. (d)

There are 2 singular value, but there are 5 singular value reported by R.

The diagonal matix are with valus $(4.828 \quad 0.828 \quad 2.083\times10^{-16} \quad 2.443\times10^{-16} \quad 1.711\times10^{-16})$.

I think the main reason is that the computation used by R involves with bidiagonal matrix, and it use the iteration to solve the SVD decomposition.

I try calculate the SVD decomposition of the matrix $M7 = \begin{pmatrix} 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \end{pmatrix}$, which has the same column space as of the matrix M6, and the rank should be 2. And I have the diagonal matix with valus $(2.921 \quad 0.689 \quad 0 \quad 0 \quad 0)$, reported by R. So I suppose it is because the iteration used by R to calculate the SVD decomposition.

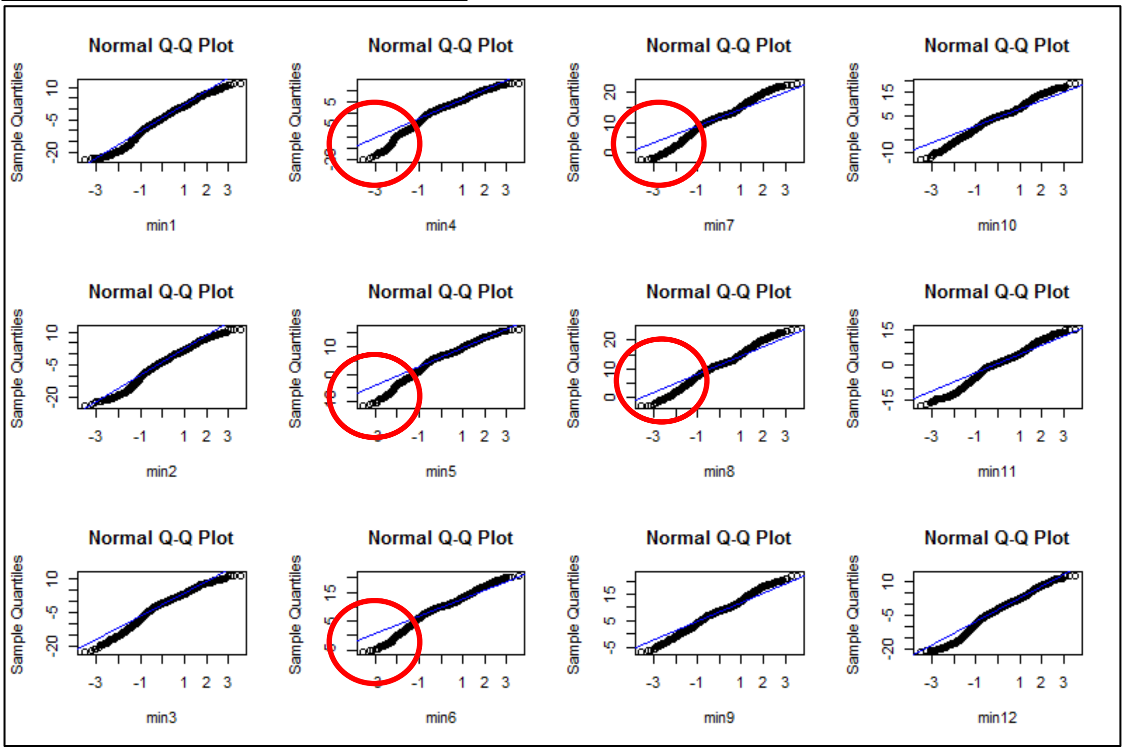
2. The SVD on Weather Data

2. (a)

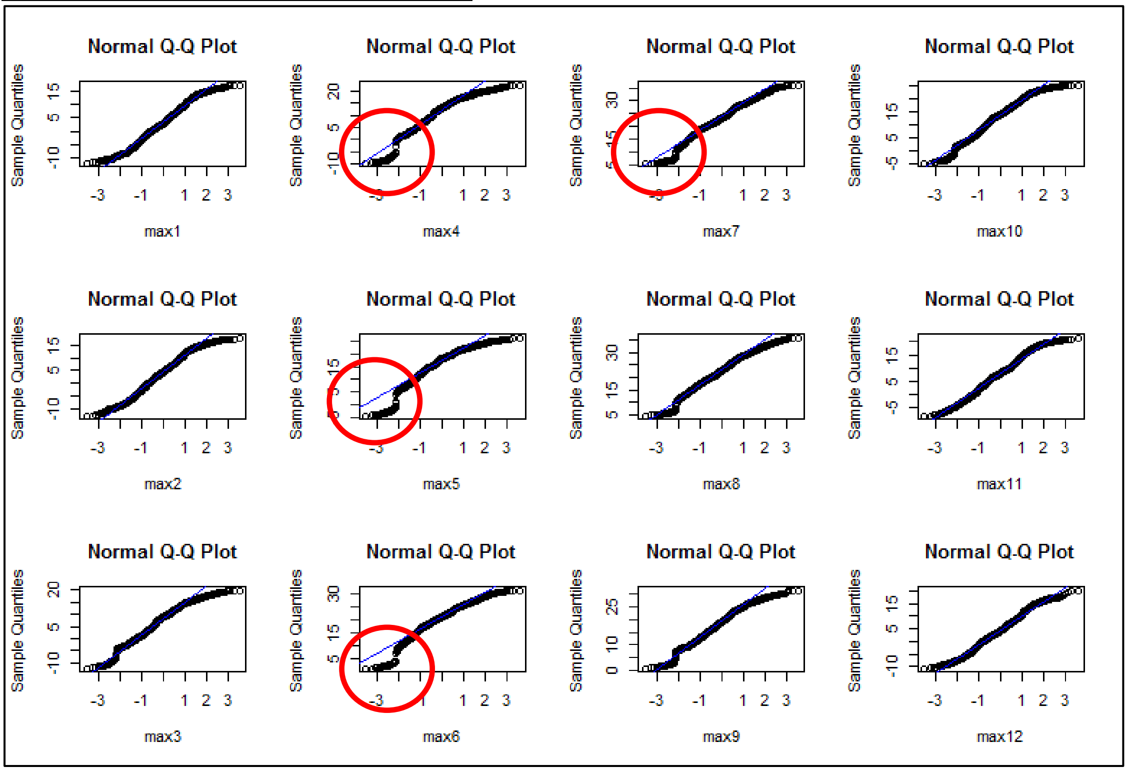
ANS:

I draw the Quantile plots as below, and circle the part that clearly does not fit the quantile line. With the Quantile plots, we can see the distribution of temperature does not quite fit the quantile line when it comes to the passage from spring to summer. For the rainfall data, the average rainfalls for each month obviously do not fit the quantile line. I would say **it is a rather weak assumption about normal distribution of the data**.

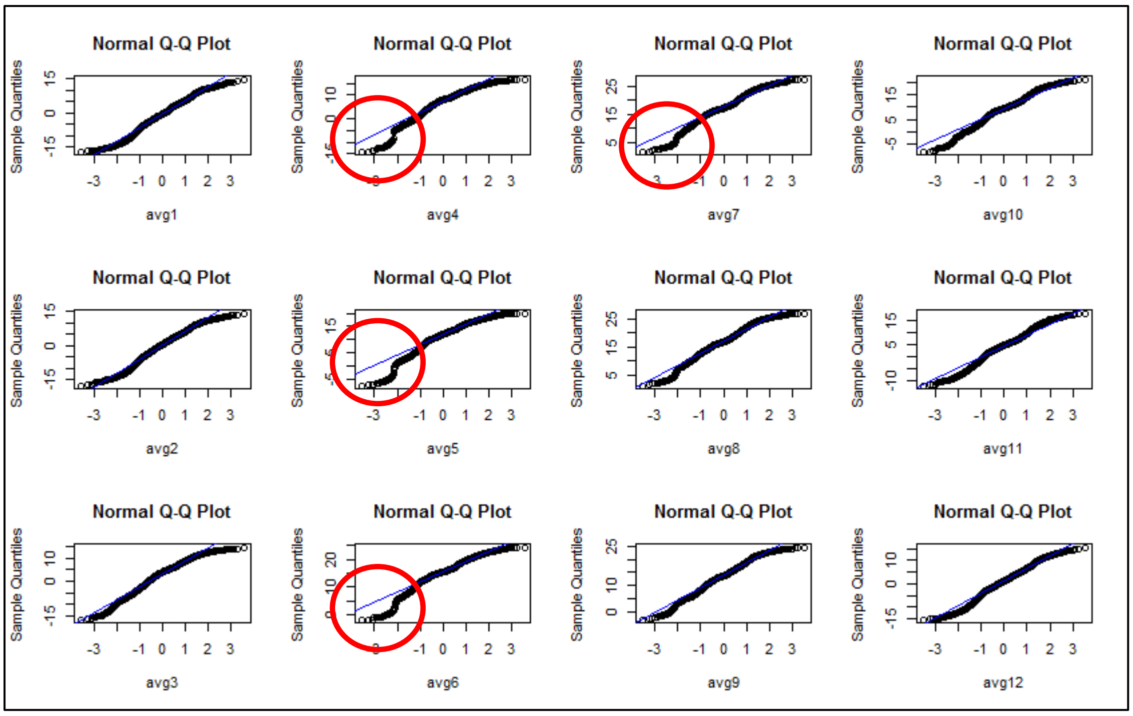
Minimum temperature for each month



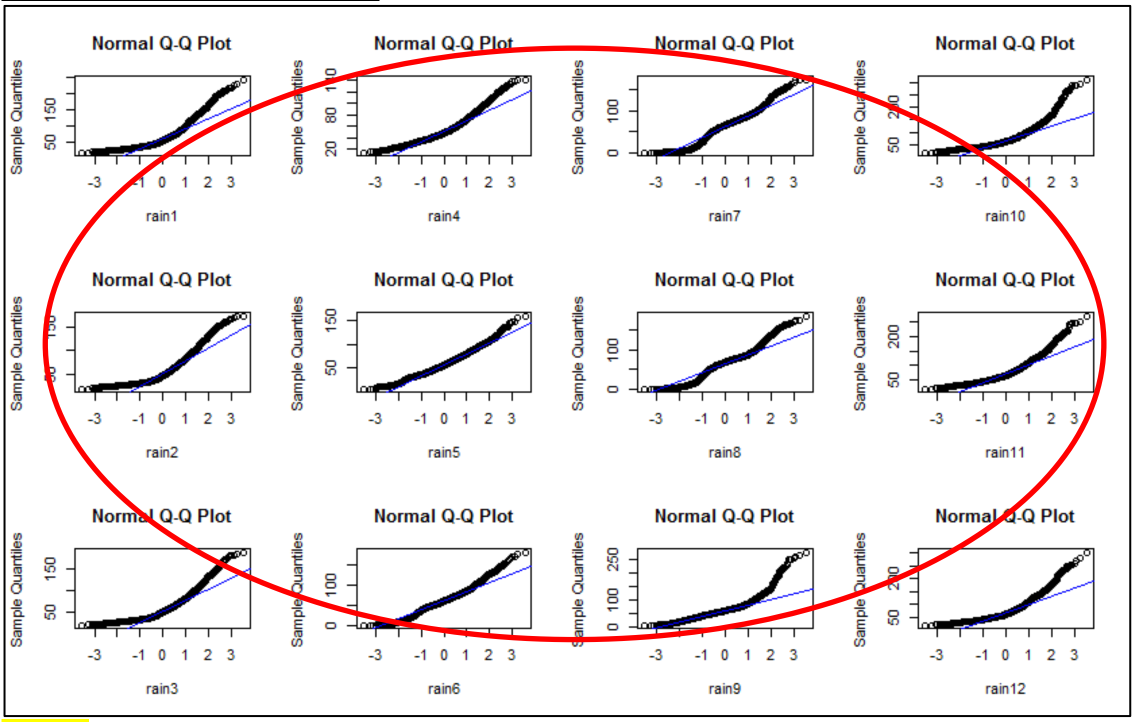
Maximum temperature for each month



Average temperature for each month



Average rainfall for each month



#code#

```
##import data
climate <- as.matrix(read.csv("C:/Users/Desktop/worldclim.csv"))
coord <- as.matrix(read.csv("C:/Users/Desktop/worldclim_coordinates.csv"))

##qq plot
par(mfcol=c(3,4))
for (i in 1:12) {qqnorm(climate[,i], xlab=colnames(climate)[i]);qqline(climate[,i],col='blue',lwd=1);}
> for (i in 12+(1:12)) {qqnorm(climate[,i], xlab=colnames(climate)[i]);qqline(climate[,i],col='blue',lwd=1);}
> for (i in 24+(1:12)) {qqnorm(climate[,i], xlab=colnames(climate)[i]);qqline(climate[,i],col='blue',lwd=1);}
> for (i in 36+(1:12)) {qqnorm(climate[,i], xlab=colnames(climate)[i]);qqline(climate[,i],col='blue',lwd=1);}
```

2. (b)

ANS:

I use the coding below to generate the SVD of the climate data. And I find the rank of the data is 48 by calculating the diagonal product of Σ.

```
#code#
##generate SVD of the data
climate.zc<-climate-matrix(rep(apply(climate,2,mean)),nrow=dim(climate)[1],ncol=dim(climate)[2],byrow=TRUE)
climate.normal<-climate.zc/matrix(rep(apply(climate,2,sd)),nrow=dim(climate)[1],ncol=dim(climate)[2],byrow=TRUE)
climate.svd<-svd(climate.normal)
##find the rank of the data by calculating the diagonal product of Σ
prod(climate.svd$d)
```

2. (c)

ANS:

I use the coding below to generate temperature plots for the value in column 1 to column 5 of matrix U. I use the columns of matrix U as the new way to explain the original data set. As the presentation indicated by the value in column 1 explains the major difference of the original data, we can generally conclude that the coorelation (or the similarity) between the sample (original data) can be roughly distinguished by "**latitude**".

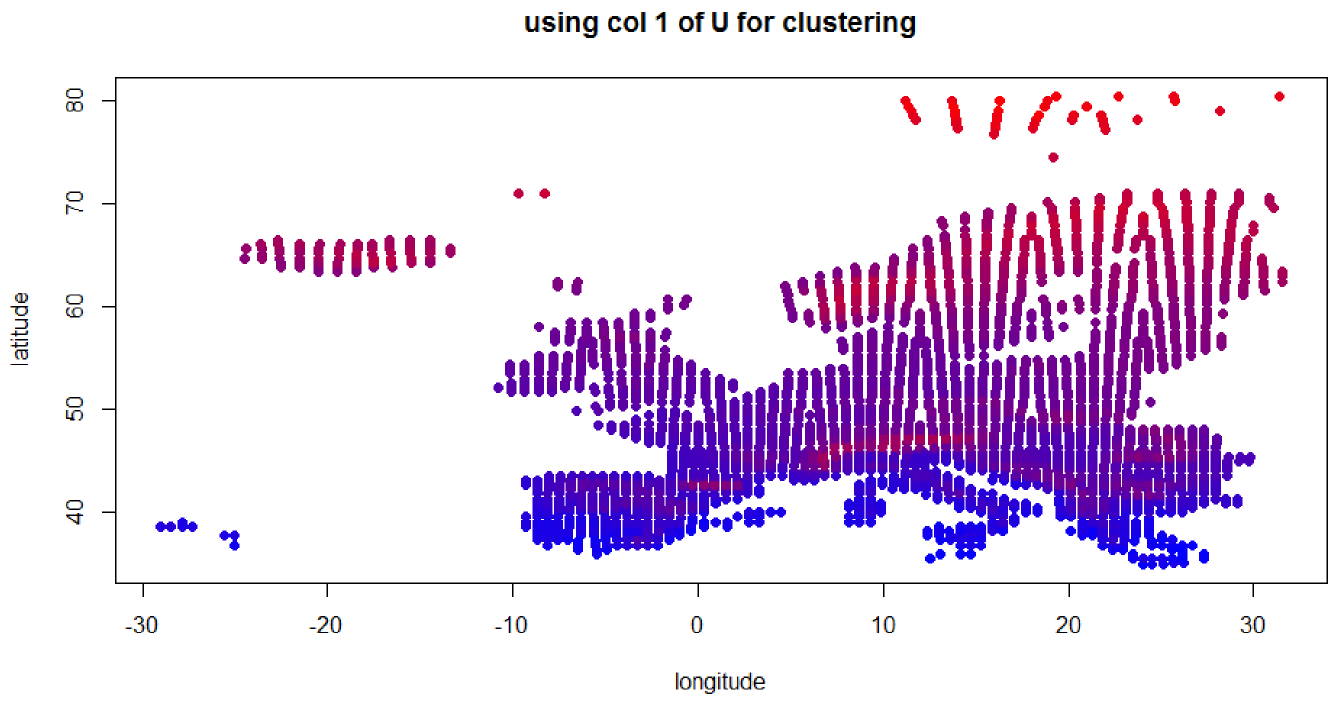
For that column 2 explains the second major difference of the original data, from the second graph, I can see the area "**near the coast**" of English Channel and the Mediterranean Sea are different from other area in a sense of second distinguishing standard.

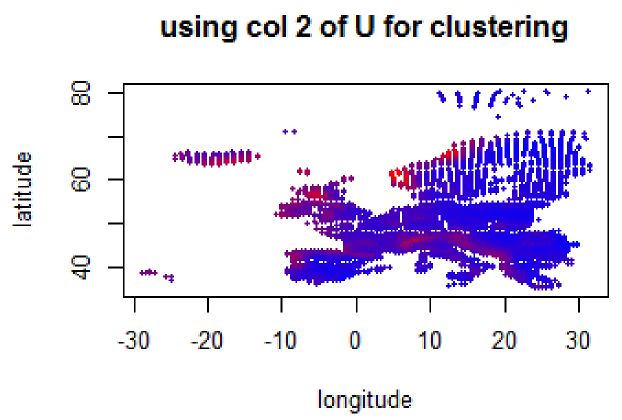
The column 3 explains the third major difference of the original data, from the second graph, I can see the area that is "**in land**" are different from other area in a sense of third distinguishing standard.

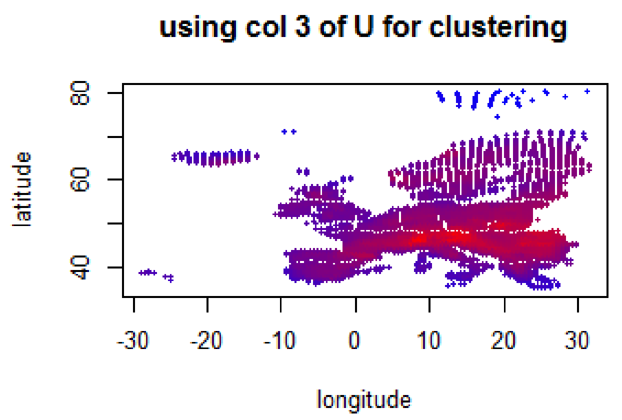
So on and so forth. And it is reasonably that it gets more blurry and harder to explain using the far right side of the singular vector of the matrix U as a distinguishing method.

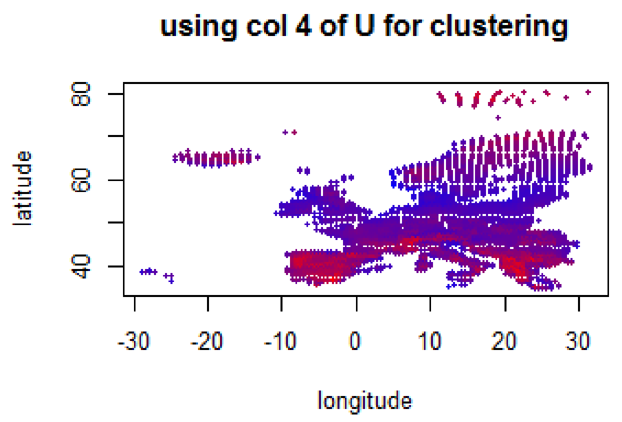
Temperature plots from Jan to May

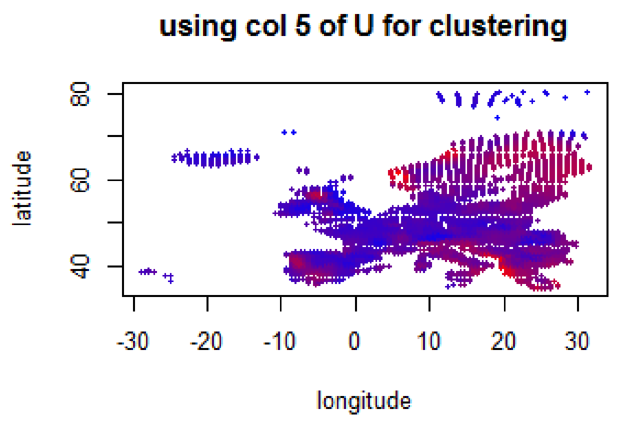
BLUE represents low values; RED represents high values











#code#

```
par(mfrow=c(2,3))
plot(coord, col=bluered(climate.svd$u[,1]),pch=19,cex=0.5,xlab='longitude',ylab='latitude',main='using col 1 of U for clustering')
plot(coord, col=bluered(climate.svd$u[,2]),pch=19,cex=0.5,xlab='longitude',ylab='latitude',main='using col 2 of U for clustering')
plot(coord, col=bluered(climate.svd$u[,3]),pch=19,cex=0.5,xlab='longitude',ylab='latitude',main='using col 3 of U for clustering')
plot(coord, col=bluered(climate.svd$u[,4]),pch=19,cex=0.5,xlab='longitude',ylab='latitude',main='using col 4 of U for clustering')
plot(coord, col=bluered(climate.svd$u[,5]),pch=19,cex=0.5,xlab='longitude',ylab='latitude',main='using col 5 of U for clustering')
```

2. (d)

ANS:

The previous information has provided me some insights:
(i). the standard 1 (col 1) are related to latitude;
(ii). the standard 2 (col 2) are related to the closeness to the sea;
(iii). the standard 3 (col 3) are related to the departure from the sea.

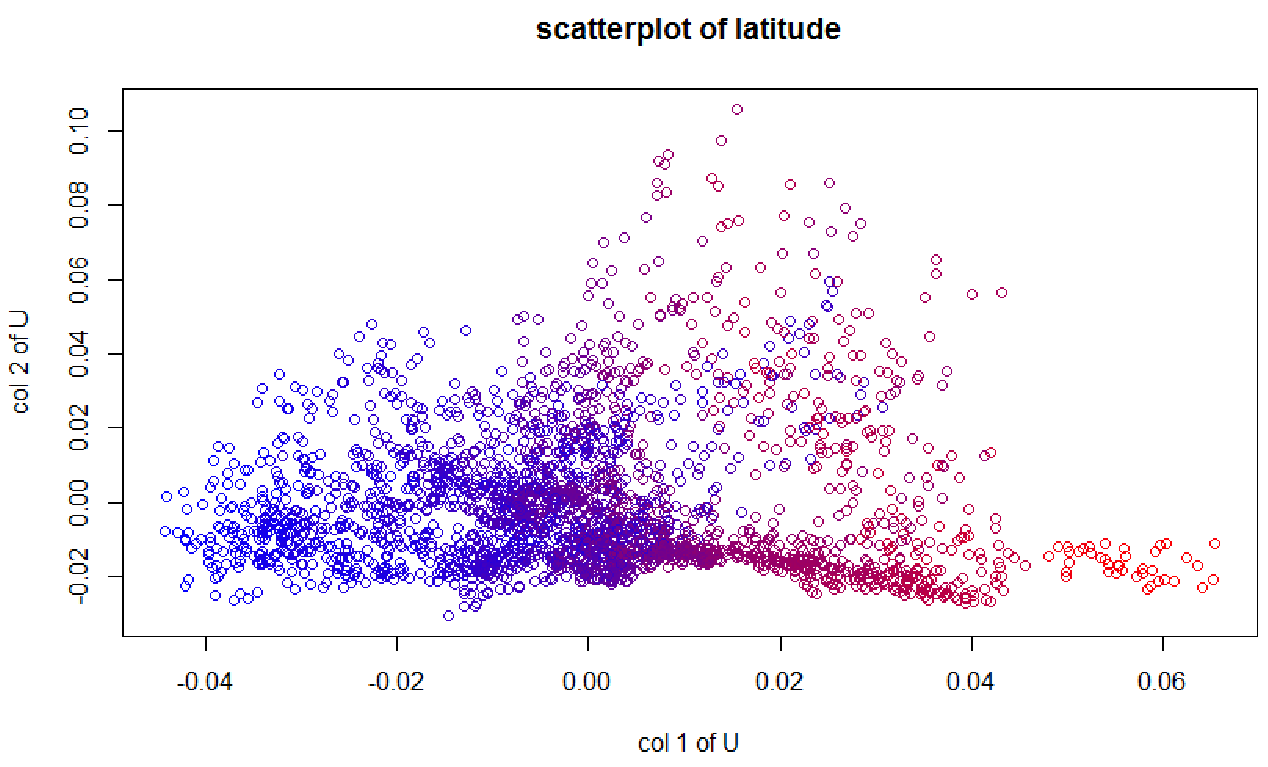
Now, I observe the graph in here.
I use the latitude as plots, so **the information of North-South has presented itself**.

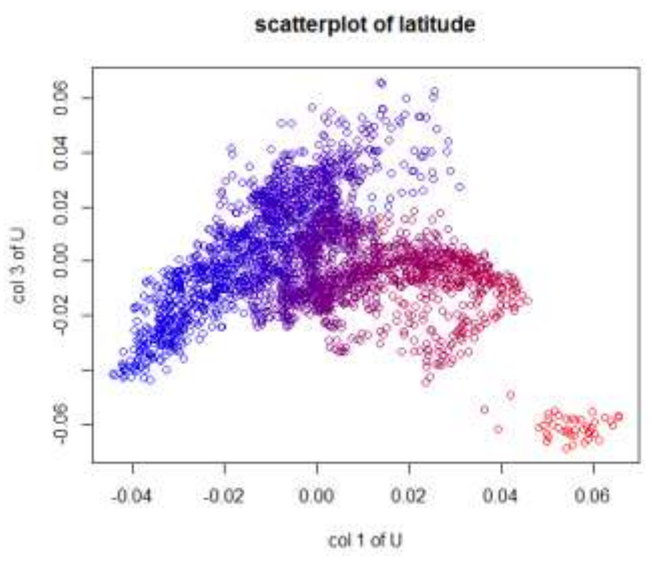
I will need to tell **the East-West information** using the graph.
My inferences:
(i) For the graph using col 1 and col 2 as axis, I will say the red dots on the upper part are probably the data from the west coast part.
(ii) For the graph using col 1 and col 3 as axis, I will say the red dots are probably the data from the inland-East part.

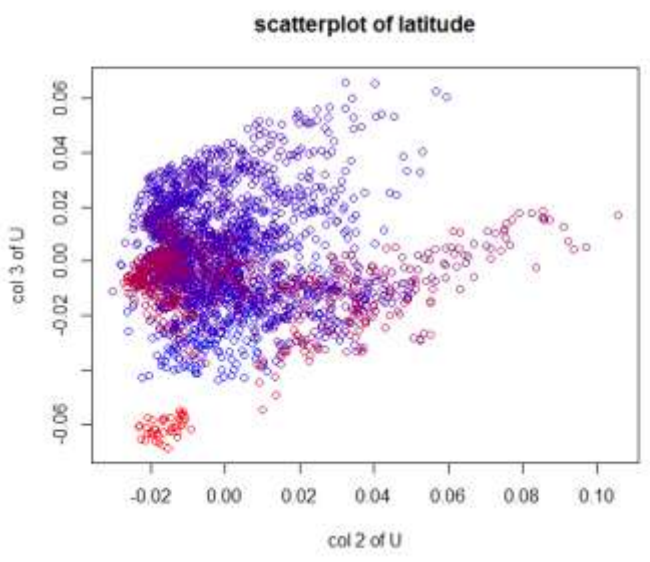
For the graph using col 2 and col 3, it is easy to predict that the plot will be more blurry, since the new axis with column 2 and 3 has less explanatory meanings than column 1.

Scatterplots using two columns of Matrix U

BLUE represents low values; RED represents high values







#code#
```
par(mfrow=c(2,2))
plot(climate.svd$u[,c(1, 2)], col=bluered(coord[,2]),xlab='col 1 of U',ylab='col 2 of U',main='scatterplot of latitude')
plot(climate.svd$u[,c(1, 3)], col=bluered(coord[,2]),xlab='col 1 of U',ylab='col 3 of U',main='scatterplot of latitude')
plot(climate.svd$u[,c(2, 3)], col=bluered(coord[,2]),xlab='col 2 of U',ylab='col 3 of U',main='scatterplot of latitude')
plot(climate.svd$u[,c(1, 4)], col=bluered(coord[,2]),xlab='col 1 of U',ylab='col 4 of U',main='scatterplot of latitude')
```

2. (e) decide what would be a good rank for a truncated SVD.

2. (e)(i) Using Guttman-Kaiser criterion

ANS: **37.**

I use the coding below to search down the position toward the far left side in the diagonal matrix Σ, where the farthest input that have not go down below 1. The result is 37.

#code#
##Search down the position toward the far left side in the diagonal matrix Σ, where the farthest input that have not go down below 1.

```
i<-1;
while(i<49){
if(climate.svd$d[i]<1){
y<-i-1;break;
}
i<-i+1;
}
>y
[1] 37
```

2. (e)(ii) Using 90% of squared Frobenius norm

ANS: **3.**

First, I generate x_i, the percentage of each squared singular value to the total sum of the squared singular values. And then I generate $F(i) = \sum_i^{48} x_i$. And I use the coding below to find the smallest number of i to reach the goal.

#code#
##generate the percentage of each squared singular value to the total sum of the squared singular values

```
prob<-climate.svd$d^2/sum(climate.svd$d^2);
i<-48;
```

##generate ddf[i] $= F(i) = \sum_i^{48} x_i$

```
ddf<-c(1:48);
while(i>0){
ddf[49-i]<-sum(prob[(49-i):48]);
i<-i-1;
}
```

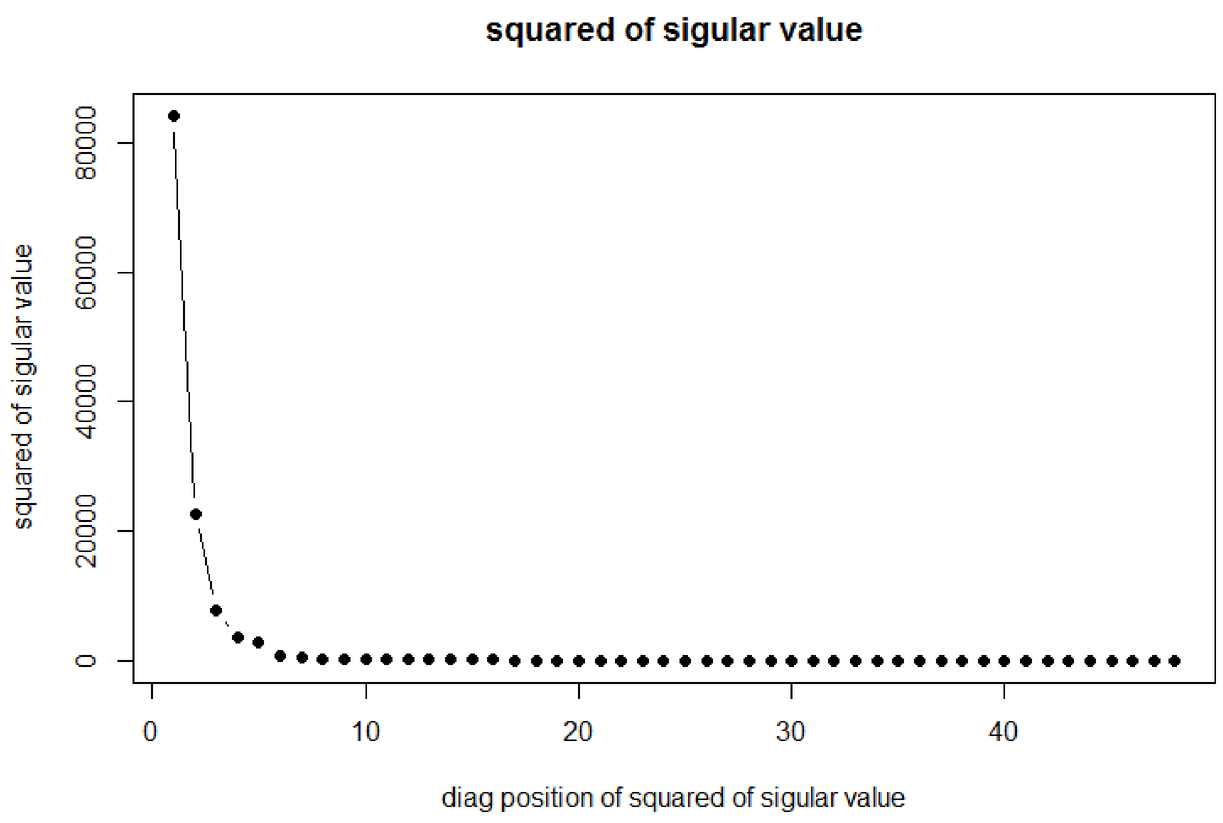
##find the rank

```
i<-1
while(i<49){
if(ddf[i]<0.1){
y<-i-1;break;
}
i<-i+1;
}
> y
[1] 3
```

2. (e)(iii) Using Scree test

ANS: **1.**

I plots the 'squared of sigular value' to 'diag position of squared of sigular value' as below. The slop drops drastically from point 1 to point 2. From a subject point of view, I would choose k as 1.



#code#

## plot the 'squared of sigular value' to 'diag position of squared of sigular value'

plot(climate.svd$d^2,pch=19, xlab='diag position of squared of sigular value',ylab='squared of sigular value', main='squared of sigular value',type='b')

2. (e)(iv) Using Entropy-based method

ANS: **1.**

First, I generate x_i, the percentage of each squared singular value to the total sum of the squared singular values. And then I generate $F(i) = \sum_i^{48} x_i$. And I calculate the entropy, which turn out to be around 0.2752. Then I use the coding below to find the smallest number of i to reach the goal, where i turn out to be 1.

#code#

##generate the percentage of each squared singular value to the total sum of the squared singular values

prob<-climate.svd$d^2/sum(climate.svd$d^2);

##generate $\mathbf{ddf[i]} = \boldsymbol{f(i)} = \sum_i^{48} \boldsymbol{x_i}$

```
i<-48;
ddf<-c(1:48);
while(i>0){
ddf[49-i]<-sum(prob[(49-i):48]);
i<-i-1;
}
```

##calculate entropy

entropy<-(-log(dim(climate)[2])^(-1))*sum(prob*log(prob))

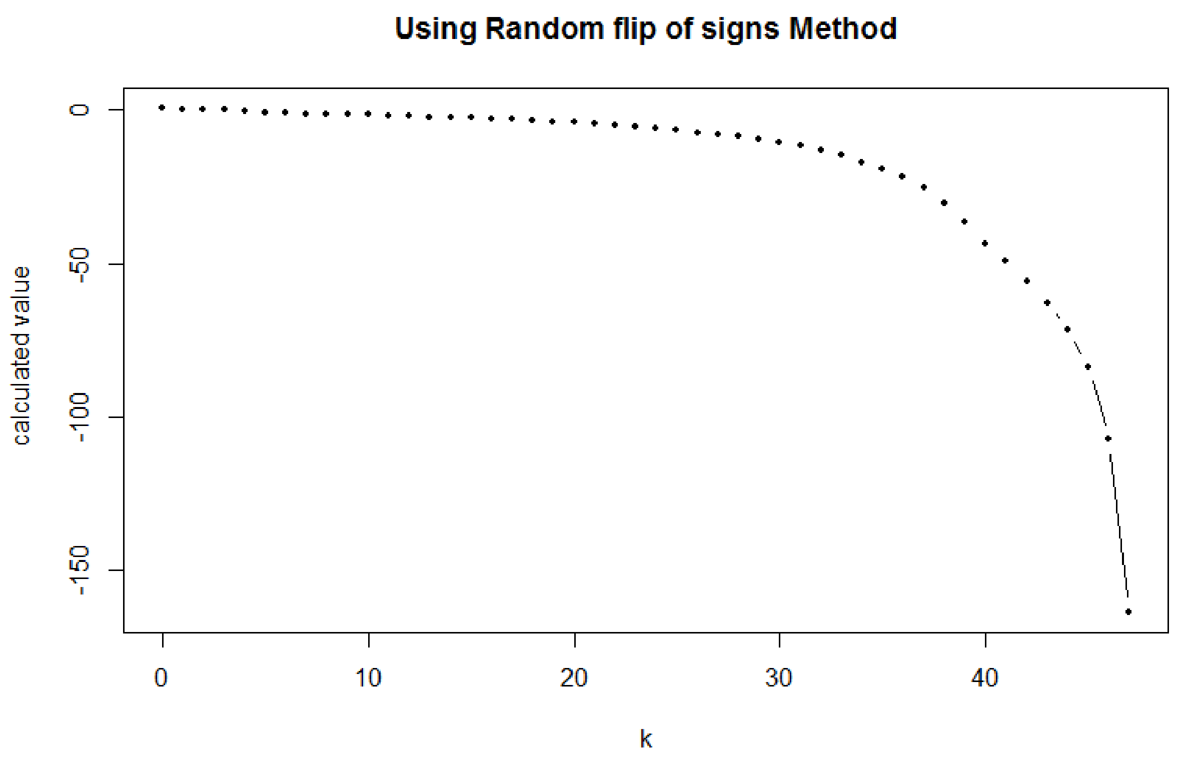
##find rank

```
i<-1;
while(i<49){if(ddf[i]<(1-entropy)){y<-i-1;break;}
i<-i+1;
}
y
```

2. (e)(v) Using Random flipping of signs

ANS: **cannot tell.**

I use the code below to generate $(\|\boldsymbol{A}_{-k}\|_2 - \|\tilde{\boldsymbol{A}}_{-k}\|_2)/\|\boldsymbol{A}_{-k}\|_F$ for k between 1 to 48, and plot the calculated value against k as below. For the purpose of finding the small value of $(\|\boldsymbol{A}_{-k}\|_2 - \|\tilde{\boldsymbol{A}}_{-k}\|_2)/\|\boldsymbol{A}_{-k}\|_F$, since it is a dropping-trend as the k increase, without the information of the threshold for the calculated value, I cannot decide which K I should choose.



#code#

##build a sequence to restore the value, for 48 cases.

```
fs<-c(1:48)
```

##generate the value $(\|\boldsymbol{A}_{-k}\|_2 - \|\tilde{\boldsymbol{A}}_{-k}\|_2)/\|\boldsymbol{A}_{-k}\|_F$, for k = 1 ~ 48

```
i<-0
while(i<48){
```

##case for using diagonal matrix Σ with single value

```
    if(i==47)
    {
```

##calculate $\|\boldsymbol{A}_{-k}\|_2$

```
        temp1<-svd(climate.svd$d[48]*climate.svd$u[,48]%*%t(climate.svd$v[,48]),0,0)$d[1];
```

## calculate $\|\tilde{\boldsymbol{A}}_{-k}\|_2$

```
        rsm<-sample(c(-1,1), nrow(climate.normal)*ncol(climate.normal), replace=T);
        mx2<-svd(rsm*climate.normal);
        temp2<-svd(mx2$d[48]*mx2$u[,48]%*%t(mx2$v[,48]),0,0)$d[1];
```

## calculate $(\|\boldsymbol{A}_{-k}\|_2 - \|\tilde{\boldsymbol{A}}_{-k}\|_2)/\|\boldsymbol{A}_{-k}\|_F$ and restore the value

```
        fs[i+1]<-(temp1-temp2)/norm(climate.svd$d[48]*climate.svd$u[,48]%*%t(climate.svd$v[,48]),'F');
        i<-i+1;
    }
    else{
        ##case for using diagonal matrix Σ with multiple value
        temp1<-svd(climate.svd$u[,c((i+1):48)]%*%diag(climate.svd$d[c((i+1):48)])%*%t(climate.svd$v[,c((i+1):4
        8)]),0,0)$d[1];
        rsm<-sample(c(-1,1), nrow(climate.normal)*ncol(climate.normal), replace=T);
        mx2<-svd(rsm*climate.normal);
        temp2<-svd(mx2$u[,c((i+1):48)]%*%diag(mx2$d[c((i+1):48)])%*%t(mx2$v[,c((i+1):48)]),0,0)$d[1];
        fs[i+1]<-(temp1-temp2)/norm(climate.svd$u[,c((i+1):48)]%*%diag(climate.svd$d[c((i+1):48)])%*%t(climate.
        svd$v[,c((i+1):48)]),'F');
        i<-i+1;
    }
}
##plot the graph to observe, and select the value
xl<-seq(0,47,1);
plot(xl,fs,pch=19,cex=0.5,type='b',xlab='k',ylab=' calculated value',main='Using Random-flip-of-signs Method');
```

2. (e) final question: personal preference

ANS: I will use 90% of squared Frobenius norm as the criterion of choosing K.

With the information of diagonal matrix Σ of SVD, I will choose the k that can explain at least 90% of squared Frobenius norm. From the statistical point of view, the explanatory of the data are strongly related to the variance of the data. I will emphasize how many information I gather that can explain the majority of the variance of the raw data, and then I will decide how many variables I should choose to achieve that goal.

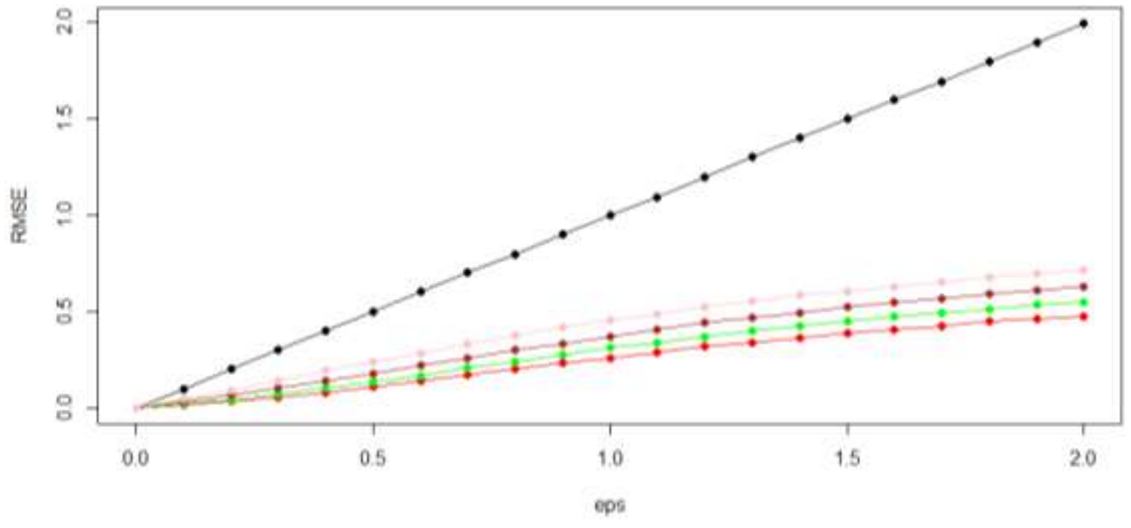
2. (f)

ANS:

I use the code to find the RMSE between the original data and the noisy data (BLACK line in the graph), and I repeat the process for specified K (k ∈ {1,2,5,10}) respectively. And I plot the graph as below. We can see that for k ∈ {1,2,5,10}, the RMSE decrease as K decreases under fixed EPS, and the RMSE increases as the EPS increases under fixed K. This represents that by choose lower k, though we might lose the information of the original data, we are less affected by noise with a certain K.

RMSE for specified K

BLACK line: RMSE between normal and noise data for K = 48; PINK line: RMSE between normal and noise data for K = 10;
BROWN line: RMSE between normal and noise data for K = 5; GREEN line: RMSE between normal and noise data for K = 2;
RED line: RMSE between normal and noise data for K = 1.



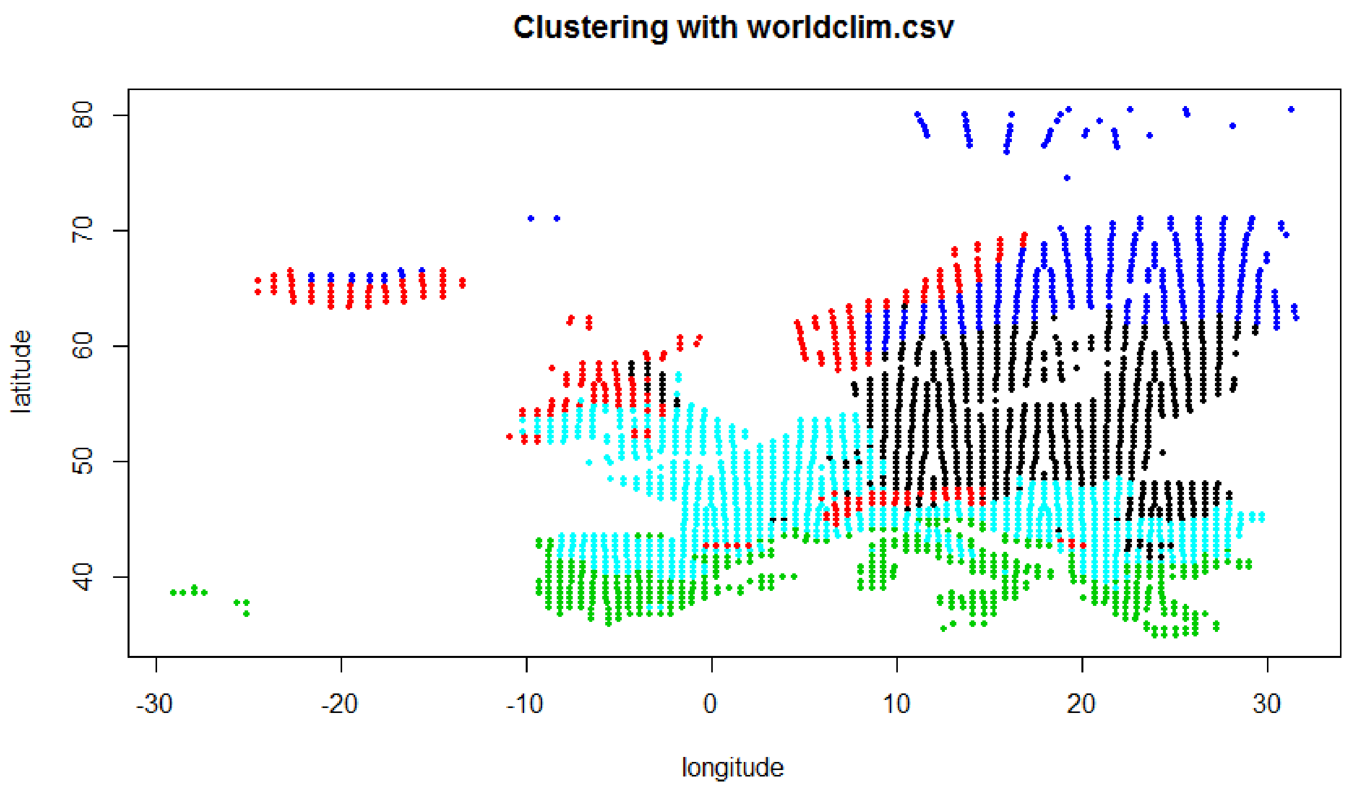
```
#code#
##set up a matrix for storage
rmse.sk<-matrix(rep(0),nrow=4,ncol=21);
##store specific K
sk<-c(1,2,5,10)
climate.svd<-svd(climate.normal);
r<-1
##fulfill the matrix
while(r<5){
##specify K
i<-sk[r];
j<-0;
while(j<21){
eps<-0.1*j;
climate.noise <- climate.normal + rnorm(prod(dim(climate.normal)))*eps;
climate.noise.zc<-climate.noise-
matrix(rep(apply(climate.noise,2,mean)),nrow=dim(climate.noise)[1],ncol=dim(climate.noise)[2],byrow=TRUE);
climate.noise.nor<-
climate.noise.zc/matrix(rep(apply(climate.noise,2,sd)),nrow=dim(climate.noise)[1],ncol=dim(climate.noise)[2],byrow
=TRUE);
climate.noise.svd<-svd(climate.noise.nor);
##calculate the RMSE for specific K using approximation of truncated original data and truncated noise data
if(i==1)
{
prox.climate.nor<-climate.svd$d[1]*climate.svd$u[,1]%*%t(climate.svd$v[,1]);
prox.climate.noise.nor<-climate.noise.svd$d[1]*climate.noise.svd$u[,1]%*%t(climate.noise.svd$v[,1])
##store the RMSE under specified K
rmse.sk[r,(j+1)]<-norm(prox.climate.nor-prox.climate.noise.nor,'F')/prod(dim(climate))^(0.5)
}else{
prox.climate.nor<-climate.svd$u[,c(1:i)]%*%diag(climate.svd$d[c(1:i)])%*%t(climate.svd$v[,c(1:i)]);
prox.climate.noise.nor<-
climate.noise.svd$u[,c(1:i)]%*%diag(climate.noise.svd$d[c(1:i)])%*%t(climate.noise.svd$v[,c(1:i)])
##store the RMSE under specified K
rmse.sk[r,(j+1)]<-norm(prox.climate.nor-prox.climate.noise.nor,'F'))/prod(dim(climate))^(0.5)
}
j<-j+1;
}
r<-r+1;
}
##plot the RMSE with specified K
rmse<-matrix(rep(0),nrow=1,ncol=21);
j<-0;
while(j<21){
eps<-0.1*j;
climate.noise <- climate.normal + rnorm(prod(dim(climate.normal)))*eps;
rmse[j+1]<-norm(climate.noise-climate.normal)/prod(dim(climate))^(0.5);
j<-j+1;
}

plot(seq(0,2,0.1),rmse,pch=19,cex=1,main='RMSE for specified
K',xlab='eps',ylab='RMSE',xlim=c(0,2),ylim=c(0,max(rmse)),type='o')
lines(seq(0,2,0.1),rmse.sk[1,],pch=19,cex=1,type='o',col='red');
lines(seq(0,2,0.1),rmse.sk[2,],pch=19,cex=1,type='o',col='green');
lines(seq(0,2,0.1),rmse.sk[3,],pch=19,cex=1,type='o',col='brown');
lines(seq(0,2,0.1),rmse.sk[4,],pch=19,cex=1,type='o',col='pink');
```

3. Clustering and visualizing

3. (a)

ANS:

Since the data contains temperature and rainfall information, so as the information used during the clustering process as a whole, I think a conservative way to express this result will be that the humidity relevance can be well presented in this graph where the places with the relative closed level are showed in the same color.



#code#

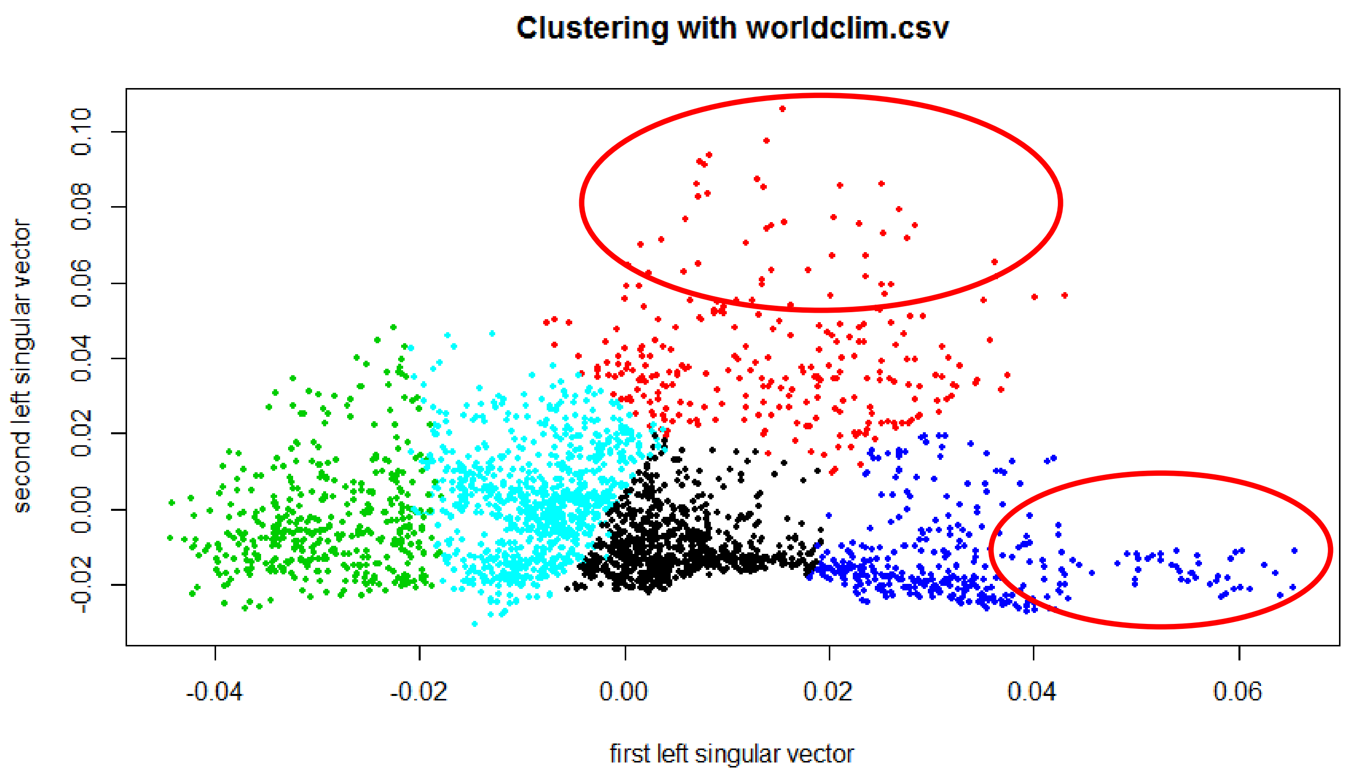
climate.cluster <- kmeans(climate.normal, 5, iter.max=100, nstart=10)$cluster;

plot(coord, col=climate.cluster,pch=19,cex=0.5,xlab='longitude',ylab='latitude',main='Clustering with worldclim.csv')

3. (b)

ANS:

In this graph, the clusters are well-separated, and the data in the red circle could be outliers.



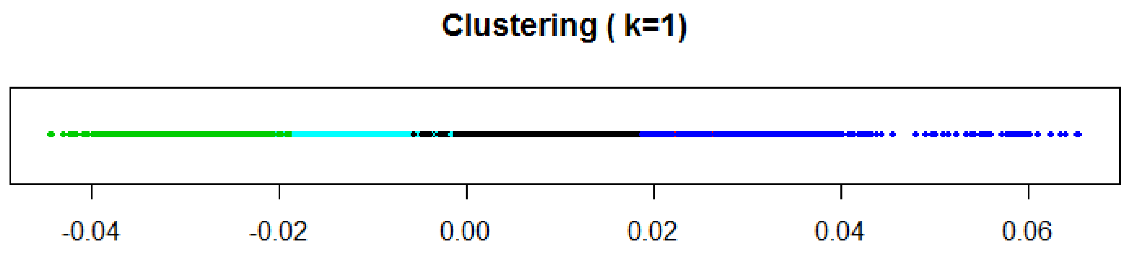
#code#
plot(climate.svd$u[,c(1,2)], col=climate.cluster,pch=19,cex=0.5,xlab='first left singular vector',ylab='second left singular vector',main='Clustering with worldclim.csv')

3. (c)

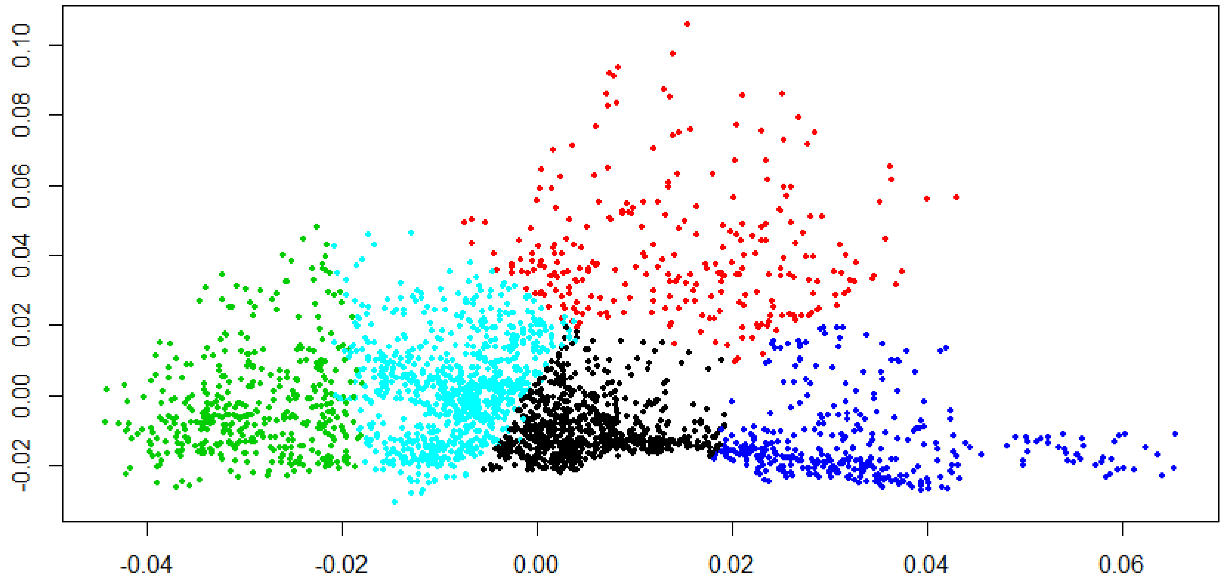
ANS:

PCA scores are as attachment. The results only differ in scale, and the pattern remains the same. The reason is that comparing to the plot using SVD, the plot of PCA is presented with the multiplication of singular values.

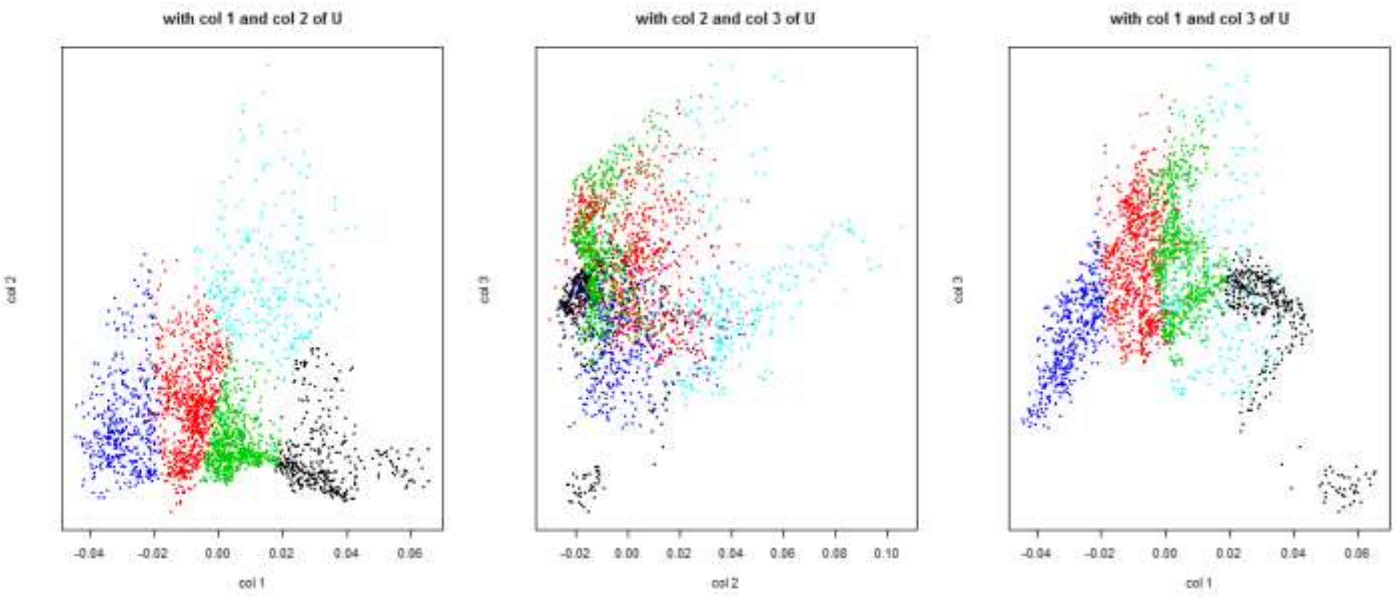
The clustering using SVD



Clustering (k=2)

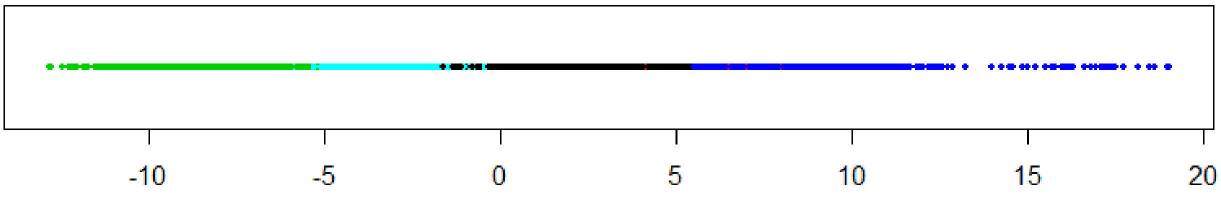


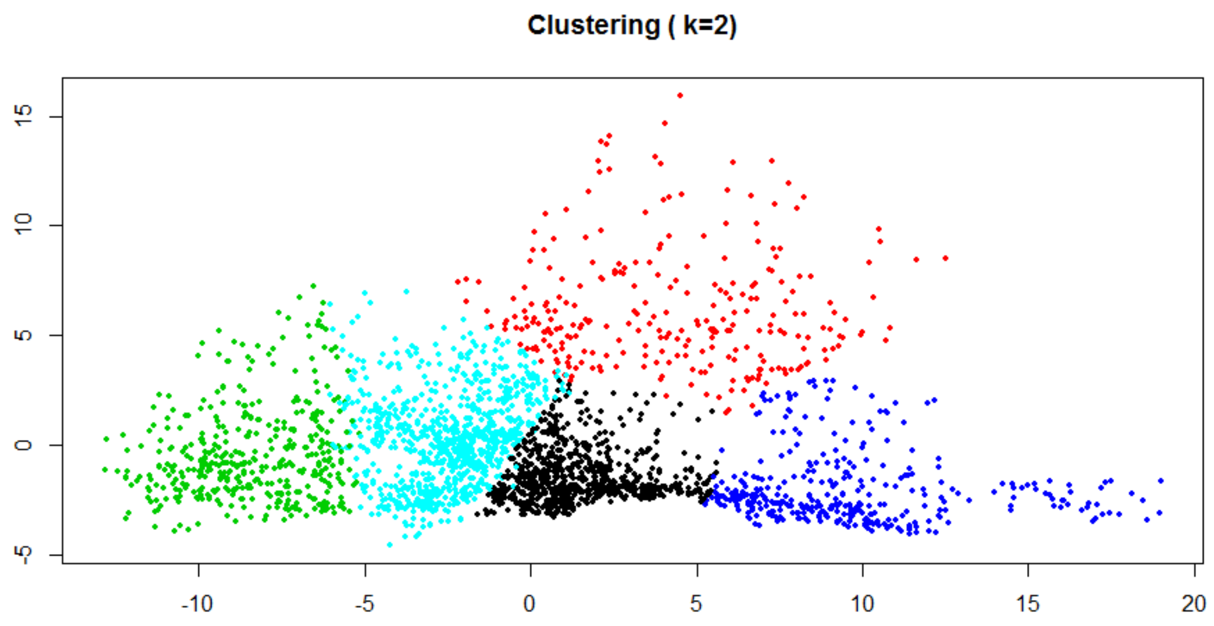
Clustering (k=3)



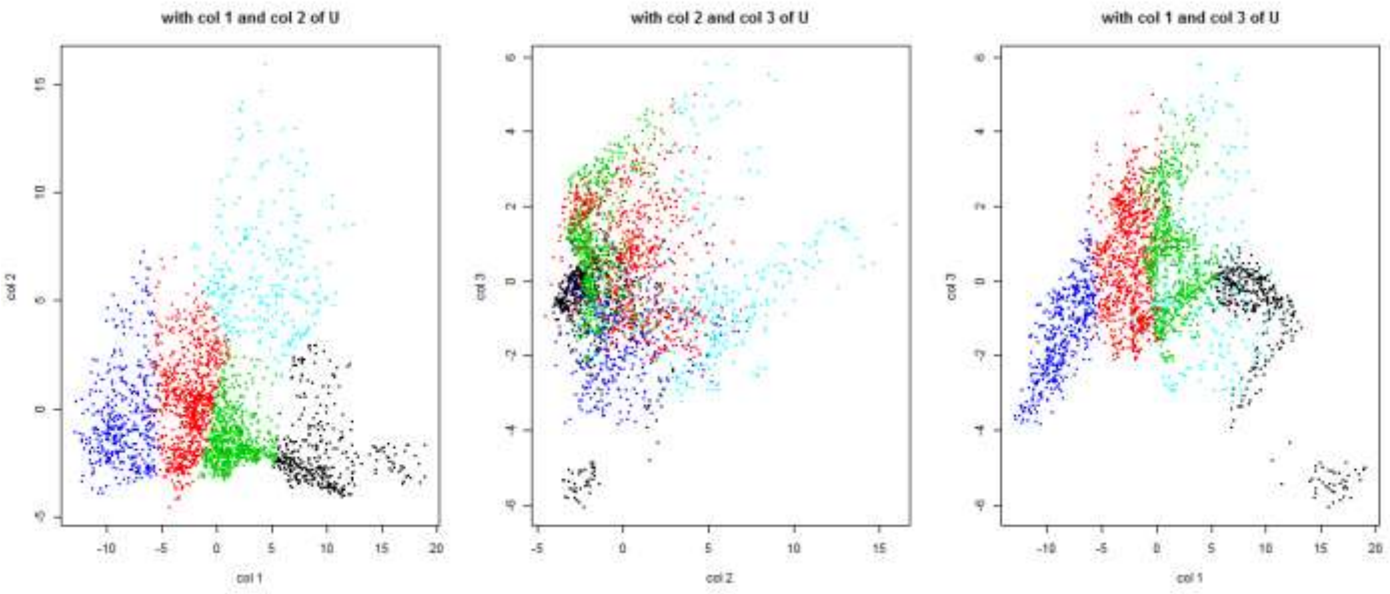
The clustering using PCA

Clustering (k=1)





Clustering (k=3)



```
#code#
##acquire all the PCA score
pca.climate<-climate.svd$u%*%diag(climate.svd$d)
##clustering using SVD with k=1
yl<-rep(0,dim(climate)[1])
plot(climate.svd$u[,1],yl, col=climate.cluster,pch=19,cex=0.5,main='Clustering ( k=1)',yaxt='n',ylab='',xlab='')
##clustering using SVD with k=2
plot(climate.svd$u[,c(1,2)], col=climate.cluster,pch=19,cex=0.5,main='Clustering ( k=2)',ylab='',xlab='')
##clustering using PCA with k=1
yl<-rep(0,dim(climate)[1])
plot(pca.climate[,1],yl, col=climate.cluster,pch=19,cex=0.5,main='Clustering ( k=1)',yaxt='n',ylab='',xlab='')
##clustering using PCA with k=2
plot(pca.climate[,c(1,2)], col=climate.cluster,pch=19,cex=0.5,main='Clustering ( k=2)',ylab='',xlab='')
```