AMS 550.400 Homework 3 Additional Instruction Due Date: Nov 19

Last Compiled on November 28, 2012

General Instruction

- failure to follow the instruction will result in severe penalty (graded at 90% or even worse)
  - no second grading is planned for this homework set
  - type up your own homework (i.e., no copy-and-pasting from others & you know one can easily check this)
- from the home directory `~/`, make a directory for this homework set
  - use `mkdir nhleeHW3.git` for the directory name
  - create a Sweave file called `nhleeHW3.Rnw`
  - replace `nhlee` with the "left-hand side" of your school email
- initialize it as a git folder
  - do this from inside of the git folder for your own sake
  - make sure to verify that your folder contains a hidden folder called `.git`
  - set it up from RStudio as a RStudio project with git support
  - look for `[COMMIT]` from the text below for the location where you are supposed to add & commit
- using RStudio for editing and compiling your Rnw file is highly recommended
  - to compile, find and press "Complie PDF" button within the RStudio editor window (typically, the upper left corner window)
  - alternatively, you can use `R CMD <Sweave/pdflatex>` from bash-shell command line provided that you are in the "appropriate" directory

```
R CMD Sweave yourfilename.Rnw
R CMD pdflatex yourfilename.tex
```

- use the `homeworkset3.tex` file as a starting point for typesetting your homework solution
  - find it from the course git folder
  - do not delete the problem statements
  - do not delete/modify the existing codes in the preamble area
- once completed, compress the folder as a `.nhleeHW3.zip` or `nhleeHW3.tar.gz`, where `nhlee` is replaced with one from your school email
  - make sure that your compressed file can actually be decompressed

- on the due date,
  - submit a paper copy to the instructor during the class
  - upload the compressed folder to the designated BB discussion forum before midnight
  - any commit made after the class meeting time will be discarded using `git reset --hard`, and will not be counted as a part of your homework submission

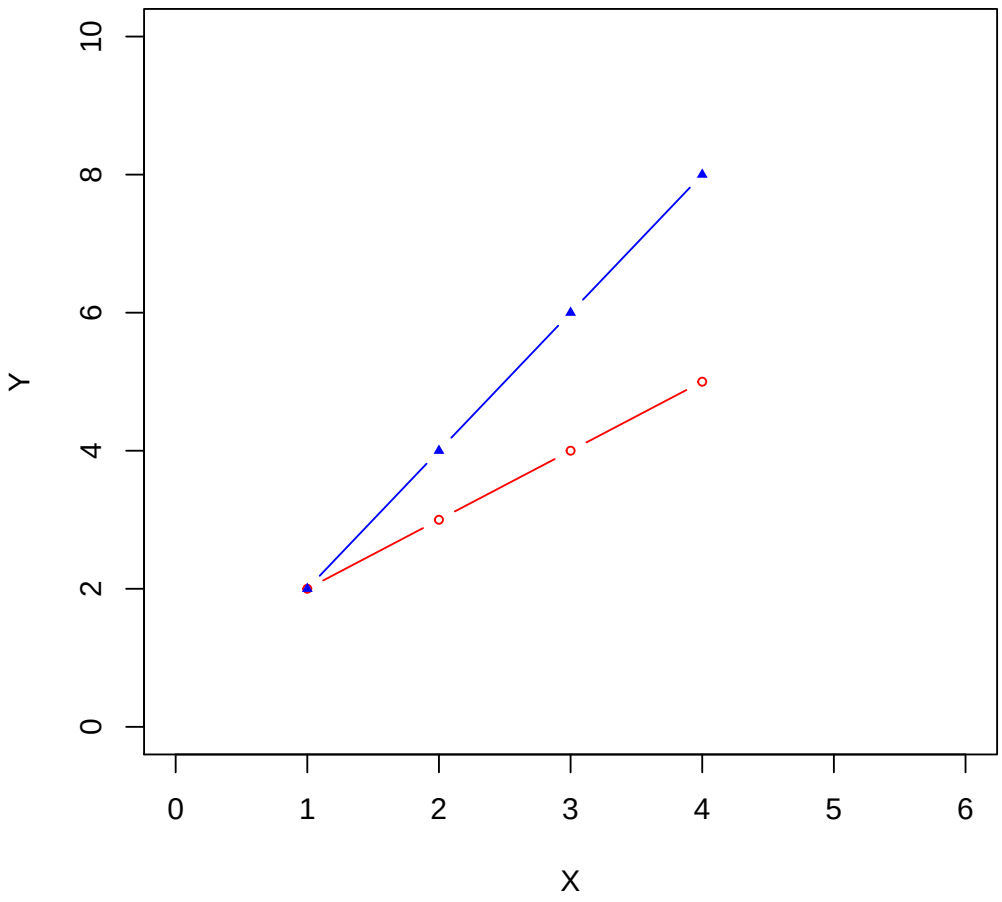
Problems from Chapter 7: Matrix Algebra for MDS

Ex 7.18

(a) `[COMMIT]` Use Sweave to accomplish this

- Use

```
#Your R codes goes here
library(Matrix)
A=matrix(c(1,2,3,2,3,4,3,4,5,4,5,6),nrow=4,byrow=TRUE)
B=matrix(c(1,2,3,2,4,6,3,6,9,4,8,12),nrow=4,byrow=TRUE)
Avec<-cbind(A[,1],A[,2])
Bvec<-cbind(B[,1],B[,2])
plot.new()
plot.window(xlim=c(0,6),ylim=c(0,10))
axis(1)
axis(2)
title(main="")
title(xlab="X")
title(ylab="Y")
box()
lines(Avec,type="b",cex=0.6,pch=21,col='red')
lines(Bvec,type="b",cex=0.6,pch=17,col='Blue')
```



- Make sure to label the horizontal axis, the vertical axis and give the main title, and give different color for A and B e.g., by filling out the space between the quotation marks, and choose a different symbol for A and B by specifying a number for `cex` and `pch`

```
plot(xydataA,xlab='',ylab='',main='',color='',cex=,pch=)
points(xydataB,cex=,pch=,cex=,color='')
```

- `[COMMIT]` Include your R codes using `lstlisting` making sure that it has an appropriate caption

(b) `[COMMIT]` Supplement your calculation using R/Sweave

- computations need to be done before using them in the text using or concurrently

```
The determinant is 1 or equivalently 1.
```

```
> A=matrix(c(1,2,3,2,3,4,3,4,5,4,5,6),nrow=4,byrow=TRUE)
> B=matrix(c(1,2,3,2,4,6,3,6,9,4,8,12),nrow=4,byrow=TRUE)
> rank_a<-qr(A)$rank
> rank_b<-qr(B)$rank
> rank_a
```

```
[1] 2

> rank_b

[1] 1
```

The rank of A is 2 or equivalently 2. The rank of B is 1 or equivalently 1.
The reason why rank of A being 2 instead of 1 is that only one of the column can be reduced. Even though, column 1 can be expressed by sum of a constant vector plus either column2 or column3, it does not fit the defination of linear dependent.

(c) [COMMIT] directly use the code and output from R/Sweave, but make sure to explain your answers
The coefficient of linear combination for Matrix A is 1:-2:1, meaning that 1*column1-2*column2+1*column
as shown below.

```
> #Combination of A
> c<-c(0,0,0)
> z<-as.matrix(c)
> a1<-cbind(z[1,],z[1,],A[,1])
> a1

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

> A1<-cbind(A[,1],A[,2],A[,3]+A[,1])
> A1

     [,1] [,2] [,3]
[1,]    1    2    4
[2,]    2    3    6
[3,]    3    4    8
[4,]    4    5   10

> a2<-cbind(z[1,],z[1,],-2*A[,2])
> a2

     [,1] [,2] [,3]
[1,]    0    0   -4
[2,]    0    0   -6
[3,]    0    0   -8
[4,]    0    0  -10

> A2<-cbind(A[,1],A[,2],A[,3]+A[,1]-2*A[,2])
> A2
```

```
     [,1] [,2] [,3]
[1,]    1    2    0
[2,]    2    3    0
[3,]    3    4    0
[4,]    4    5    0
```

The coefficient of linear combination for Matrix B is -1:-1:1, meaning that -1*column1-1*column2+1*column3=0, as shown below.

```
> #Combination of B
> b1<-cbind(z[1,],z[1,],-B[,1])
> b1

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

> B1<-cbind(B[,1],B[,2],B[,3]-B[,1])
> B1

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

> b2<-cbind(z[1,],z[1,],-B[,2])
> b2

     [,1] [,2] [,3]
[1,]    0    0   -2
[2,]    0    0   -4
[3,]    0    0   -6
[4,]    0    0   -8

> B2<-cbind(B[,1],B[,2],B[,3]-B[,1]-B[,2])
> B2

     [,1] [,2] [,3]
[1,]    1    2    0
[2,]    2    4    0
[3,]    3    6    0
[4,]    4    8    0
```

Ex 7.24

- [COMMIT] use `lstlisting` to list your R code

```
AT<-t(A)
B<-A%*%AT
c<-diag(B)
One<-matrix(c(1,1,1,1),byrow=T)
D2<-c%*%t(One)+One%*%c-2*B
D<-sqrt(D2)
print(D)
```

- [COMMIT] use R/Sweave for computation, but do not use the built-in `dist` function

```
> AT<-t(A)
> B<-A%*%AT
> c<-diag(B)
> One<-matrix(c(1,1,1,1),byrow=T)
> D2<-c%*%t(One)+One%*%c-2*B
> D<-sqrt(D2)
> print(D)

         [,1]     [,2]     [,3]     [,4]
[1,] 0.000000 1.732051 3.464102 5.196152
[2,] 1.732051 0.000000 1.732051 3.464102
[3,] 3.464102 1.732051 0.000000 1.732051
[4,] 5.196152 3.464102 1.732051 0.000000
```

- [COMMIT] use R/Sweave for computation, and this time, do use the built-in `dist` function for comparison

```
> dist(A)

         1        2        3
2 1.732051
3 3.464102 1.732051
4 5.196152 3.464102 1.732051
```

- [COMMIT] make sure to explain your computation, e.g., compare the two computations

The two method get the same result.Using equation generates the Euclidean distances and mathes the built in function.

Ex 7.30 Omit (c), (d) and (e). The necessary data is saved in `matlabclown.RData` and can be found from the course git folder. The followings are the equivalent R version:

```
load('matlabclown.RData')
image(X) # omit this in your Sweave code
svdX = svd(X)
U = svdX$u
S = diag(svdX$d)
```

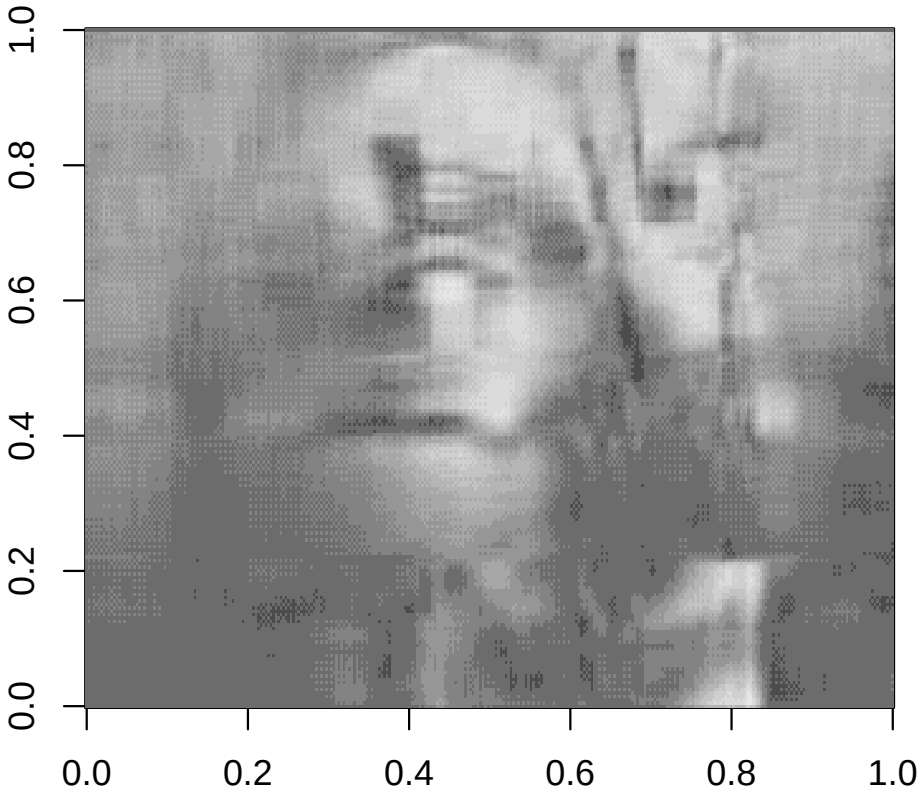
```
V = svdX$v
k = 10
M = U[,1:k,drop=FALSE] %*% S[1:k,1:k,drop=FALSE] %*% t(V[,1:k,drop=FALSE])
image(M) # omit this in your Sweave code
image(M,col=gray.colors(k))
```

(a) [COMMIT] choose a small, a medium and a large value for k

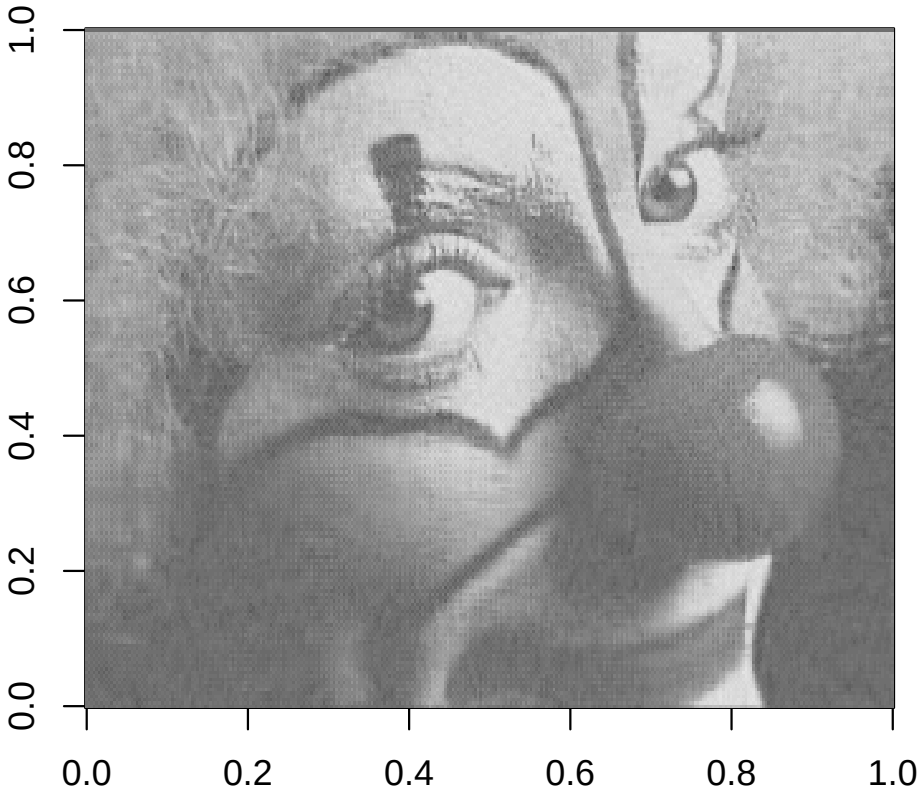
- for each k,
  - do [COMMIT]
  - your performance evaluation is to be included as a caption, and change `tinyK`, `width` and `height` accordingly

```
\begin{figure}
    \centering
\begin{Schunk}
\begin{Sinput}
> tinyK = 1
> #smallK = 5
> #mediumK = 10
> #largeK = 20
> #Your R codes go here
\end{Sinput}
\end{Schunk}
\includegraphics{ywu67HW3-008}
\caption{<YOUR PERFORMANCE EVALUATION> on $1$}
    \label{fig:matlabclownKaNumber}
\end{figure}
```

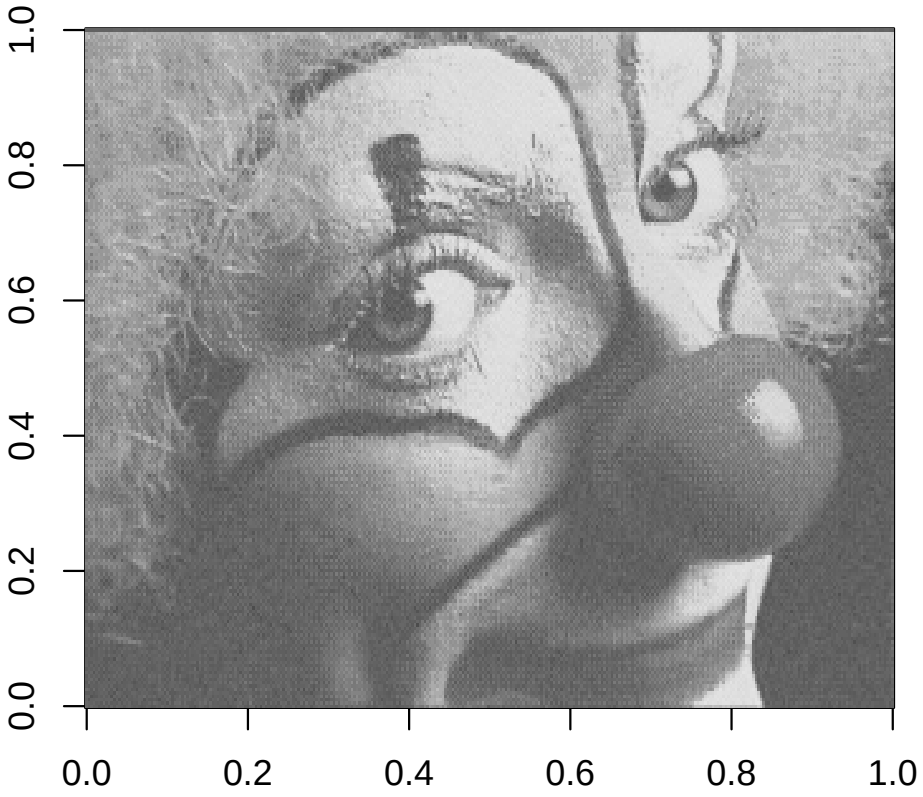
```
> #Test of small K.
> #smallK = 10
> load('matlabclown.RData')
> svdX = svd(X)
> U = svdX$u
> S = diag(svdX$d)
> V = svdX$v
> k = 10
> M = U[,1:k,drop=FALSE] %*% S[1:k,1:k,drop=FALSE] %*% t(V[,1:k,drop=FALSE])
> image(M,col=gray.colors(k))
```




```
> #Test of middle K
> #mediumK = 50
> load('matlabclown.RData')
> svdX = svd(X)
> U = svdX$u
> S = diag(svdX$d)
> V = svdX$v
> k = 50
> M = U[,1:k,drop=FALSE] %*% S[1:k,1:k,drop=FALSE] %*% t(V[,1:k,drop=FALSE])
> image(M,col=gray.colors(k))
```




```
> #Test of large K
> #largeK = 100
> load('matlabclown.RData')
> svdX = svd(X)
> U = svdX$u
> S = diag(svdX$d)
> V = svdX$v
> k = 100
> M = U[,1:k,drop=FALSE] %*% S[1:k,1:k,drop=FALSE] %*% t(V[,1:k,drop=FALSE])
> image(M,col=gray.colors(k))
```



(b) – [COMMIT] code up all your computation using R/Sweave before starting to type your explanation

```
> smallK = 10
> mediumK = 50
> largeK = 100
> load('matlabclown.RData')
> dimX= dim(X)
> k=smallK
> Compsmall=((k*(dimX[1]+dimX[2]+1)))/(dimX[1]*dimX[2])
> Compsmall
[1] 0.08140625
> k=mediumK
> Compmiddle=((k*(dimX[1]+dimX[2]+1)))/(dimX[1]*dimX[2])
> Compmiddle
[1] 0.4070313
> k=largeK
> Complarge=((k*(dimX[1]+dimX[2]+1)))/(dimX[1]*dimX[2])
```

```
> Complarge
[1] 0.8140625
```

– [COMMIT] write your explanation referring to the numbers computed in the previous step, using

Compute the compression ratio by divide the sum of dimension after compression by the sum of dimension before compression. So when k equals to 10, the ratio is 0.08140625.
When k equals to 50, the ratio is 0.40703125.
When k equals to 100, the ratio is 0.8140625.

Problems from Chapter 4: Multidimensional Scaling

Ex 4.1

- [COMMIT] Modify the code in Listing 1 for illustrating the first ten objects on a "line"

Listing 1: TikZ Code for Figure 1

```
\begin{tikzpicture}
  \foreach \x in {1,2,...,5,7,8,...,12}
    \foreach \y in {1,...,5}
    {
      \draw (\x,\y) +(-.5,-.5) rectangle ++(.5,.5);
      \draw (\x,\y) node{\footnotesize $(\x,\y)$};
    }
\end{tikzpicture}
```

- [COMMIT] list your R/Sweave codes using `lstlisting`

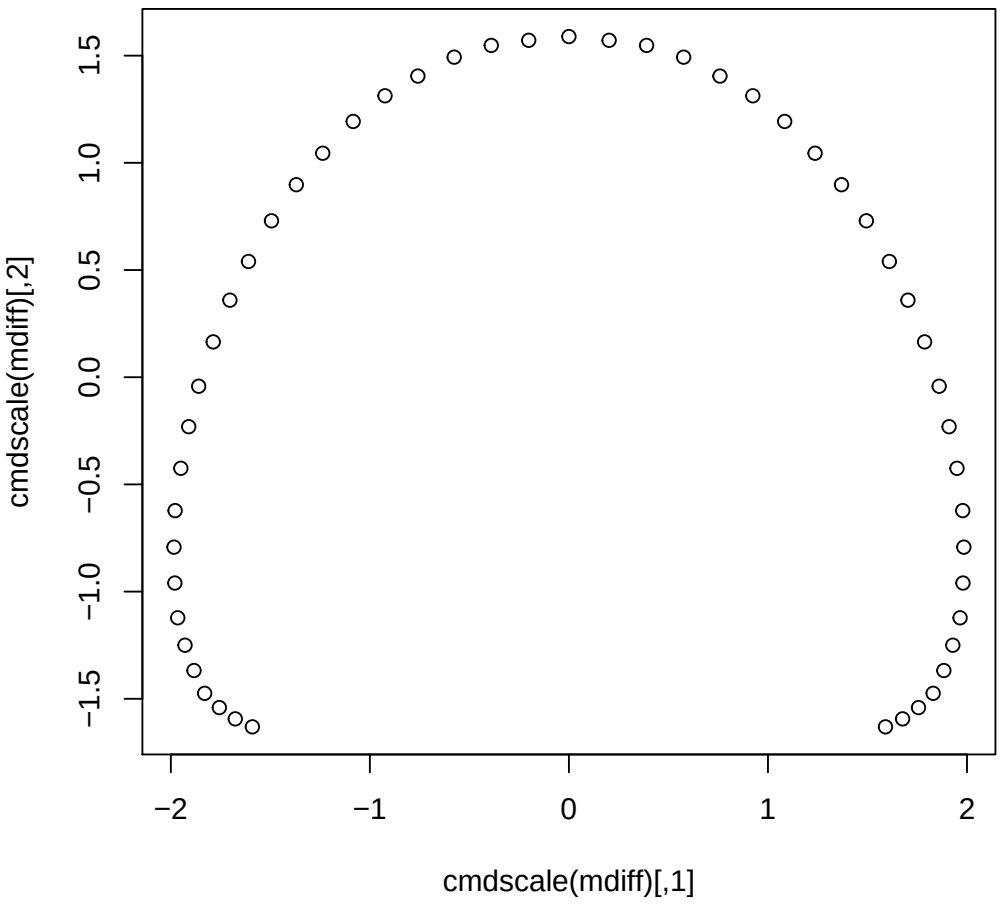
```
a <- c()
for(i in 1:51) {
  for(j in 1:51){
  if(i==j)
    {
      a=append(a,9)
    }
  else if((abs(i-j)>=1) && (abs(i-j)<=3)){
      a=append(a,8)
    }
  else if((abs(i-j)>=4)&&(abs(i-j)<=6)){
      a=append(a,7)
    }
  else if((abs(i-j)>=7)&&(abs(i-j)<=9)){
      a=append(a,6)
    }
  else if((abs(i-j)>=10)&&(abs(i-j)<=12)){
      a=append(a,5)
    }
  else if((abs(i-j)>=13)&&(abs(i-j)<=15)){
      a=append(a,4)
    }
```

```
  else if((abs(i-j)>=16)&&(abs(i-j)<=18)){
      a=append(a,3)
    }
  else if((abs(i-j)>=19)&&(abs(i-j)<=21)){
      a=append(a,2)
    }
  else if((abs(i-j)>=22)&&(abs(i-j)<=24)){
      a=append(a,1)
    }
  else if((abs(i-j)>=25)){
      a=append(a,0)
    }
  }
}
q=matrix(a,nrow=51)
for(i in 1:51){
  for(j in 1:51){
    if(i==1 && j==1){
      diff=sqrt(q[i,i]+q[j,j]+-2*q[i,j])
    } else{
      diff = append(diff, sqrt(q[i,i]+q[j,j]+-2*q[i,j]))
    }
  }
}
mdiff = matrix(diff, nrow = 51)
plot(cmdscale(mdiff))
```

- [COMMIT] use the R/Sweave codes to compute

```
> a <- c()
> for(i in 1:51) {
+   for(j in 1:51){
+   if(i==j)
+     {
+       a=append(a,9)
+     }
+   else if((abs(i-j)>=1) && (abs(i-j)<=3)){
+       a=append(a,8)
+     }
+   else if((abs(i-j)>=4)&&(abs(i-j)<=6)){
+       a=append(a,7)
+     }
+   else if((abs(i-j)>=7)&&(abs(i-j)<=9)){
+       a=append(a,6)
+     }
+   else if((abs(i-j)>=10)&&(abs(i-j)<=12)){
+       a=append(a,5)
+     }
+   else if((abs(i-j)>=13)&&(abs(i-j)<=15)){
+       a=append(a,4)
+     }
+   else if((abs(i-j)>=16)&&(abs(i-j)<=18)){
```

```
+       a=append(a,3)
+     }
+   else if((abs(i-j)>=19)&&(abs(i-j)<=21)){
+       a=append(a,2)
+     }
+   else if((abs(i-j)>=22)&&(abs(i-j)<=24)){
+       a=append(a,1)
+     }
+   else if((abs(i-j)>=25)){
+       a=append(a,0)
+     }
+   }
+ }
> q=matrix(a,nrow=51)
> for(i in 1:51){
+   for(j in 1:51){
+     if(i==1 && j==1){
+       diff=sqrt(q[i,i]+q[j,j]+-2*q[i,j])
+     } else{
+       diff = append(diff, sqrt(q[i,i]+q[j,j]+-2*q[i,j]))
+     }
+   }
+ }
> mdiff = matrix(diff, nrow = 51)
> plot(cmdscale(mdiff))
```



- `[COMMIT]` explain your computed numerical values
- make sure to refer to your R code listing via `\ref` and to the computed values using `\Sexpr`
- refer to [**?**] if necessary The

$(1,5)$	$(2,5)$	$(3,5)$	$(4,5)$	$(5,5)$
$(1,4)$	$(2,4)$	$(3,4)$	$(4,4)$	$(5,4)$
$(1,3)$	$(2,3)$	$(3,3)$	$(4,3)$	$(5,3)$
$(1,2)$	$(2,2)$	$(3,2)$	$(4,2)$	$(5,2)$
$(1,1)$	$(2,1)$	$(3,1)$	$(4,1)$	$(5,1)$

$(7,5)$	$(8,5)$	$(9,5)$	$(10,5)$	$(11,5)$	$(12,5)$
$(7,4)$	$(8,4)$	$(9,4)$	$(10,4)$	$(11,4)$	$(12,4)$
$(7,3)$	$(8,3)$	$(9,3)$	$(10,3)$	$(11,3)$	$(12,3)$
$(7,2)$	$(8,2)$	$(9,2)$	$(10,2)$	$(11,2)$	$(12,2)$
$(7,1)$	$(8,1)$	$(9,1)$	$(10,1)$	$(11,1)$	$(12,1)$

Figure 1: An extension of an example from the TikZ & PGF manual [**?**]

Ex 4.2

- [COMMIT] list your R code using `lstlisting`
- [COMMIT] demonstrate that your function is "functioning" by way of R/Sweave

Ex 4.3

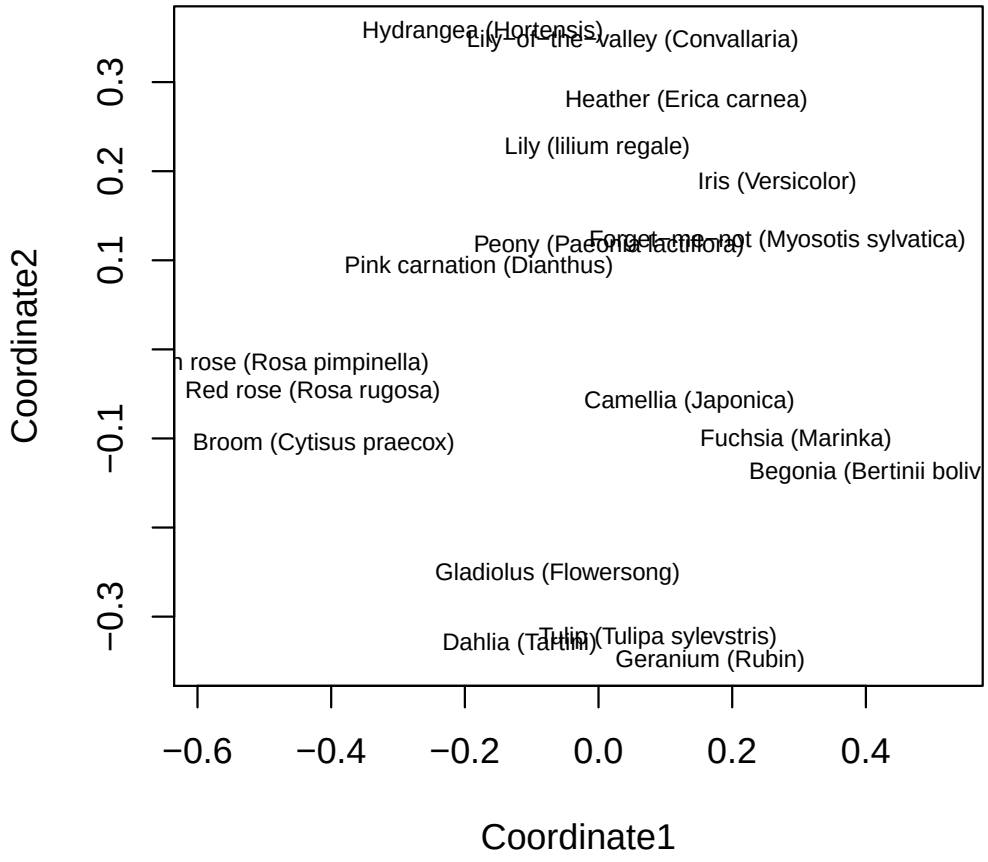
- [COMMIT] list your R code using `lstlisting`

```
require(MVA);data(gardenflowers)
mds<-cmdscale(gardenflowers,k=17,eig=T)
mds$eig
#there are negative numbers in eigenvalues
cumsum(abs(mds$eig))/sum(abs(mds$eig))
cumsum((mds$eig)^2)/sum((mds$eig)^2)
x<-mds$points[,1]
y<-mds$points[,2]
co<-cbind(x,y)
plot(co,xlab="Coordinate1",ylab="Coordinate2",xlim=range(x)*1.2  ,type='n')
text(co,rownames(co),cex=0.7)
```

- [COMMIT] load the data (`require(MVA);data(gardenflowers)`) and compute using R/Sweave

```
>     require(MVA);data(gardenflowers)
>     mds<-cmdscale(gardenflowers,k=17,eig=T)
>     mds$eig

 [1]  1.173085e+00  8.944353e-01  5.732009e-01  4.843006e-01  2.638516e-01
 [6]  2.298165e-01  8.383417e-02  6.645861e-02  2.984017e-02 -8.326673e-17
[11] -2.147959e-02 -4.094094e-02 -4.366468e-02 -8.867401e-02 -1.045334e-01
[16] -1.275889e-01 -1.714295e-01 -2.045124e-01

>     #there are negative numbers in eigenvalues
>     cumsum(abs(mds$eig))/sum(abs(mds$eig))

 [1] 0.2549273 0.4493002 0.5738645 0.6791096 0.7364481 0.7863903 0.8046086
 [8] 0.8190510 0.8255356 0.8255356 0.8302034 0.8391005 0.8485894 0.8678594
[15] 0.8905760 0.9183028 0.9555567 1.0000000

>     cumsum((mds$eig)^2)/sum((mds$eig)^2)

 [1] 0.4611160 0.7291863 0.8392805 0.9178730 0.9412006 0.9588982 0.9612532
 [8] 0.9627331 0.9630315 0.9630315 0.9631861 0.9637478 0.9643866 0.9670214
[15] 0.9706829 0.9761377 0.9859851 1.0000000
```

- [COMMIT] include a plot of (relative) positions using R/Sweave



- `[COMMIT]` allocate at least a quarter page of *text* explaining the result

Among the 18 flowers, there are generally 4 goups are more similar to each other according to the plot.
Firstly, 3 types of flowers below are close to each other in the plot, indicating more common propeties, they are:
Scotch rose (Rosa pimpinella)
Red rose (Rosa rugosa)
Broom (Cytisus praecox).
The second group which are also very similar to each others are:
Camellia (Japonica)
Fuchsia (Marinka)
Begonia (Bertinii bolivienis).
The third group contains 4 kinds of flowers sharing more common properties, they are:
Dahlia (Tartini)
Geranium (Rubin)
Gladiolus (Flowersong)
Tulip (Tulipa sylevstris)

The last group is bigger, and relatively not as clustered as the previous 3 groups. There are also subgroups in the last group.All the flowers in the group are:
Hydrangea (Hortensis)
Lily-of-the-valley (Convallaria)
Heather (Erica carnea)
Lily (lilium regale)
Iris (Versicolor)
Peony (Paeonia lactiflora)
Forget-me-not (Myosotis sylvatica)
Pink carnation (Dianthus).
Among which Hydrangea (Hortensis) and Lily-of-the-valley (Convallaria) are more similar than others;
Heather (Erica carnea), Lily (lilium regale) and Iris (Versicolor) are relatively more similar;
Peony (Paeonia lactiflora), Forget-me-not (Myosotis sylvatica) and Pink carnation (Dianthus) are closer.

References