Non-linear dimensionality reduction

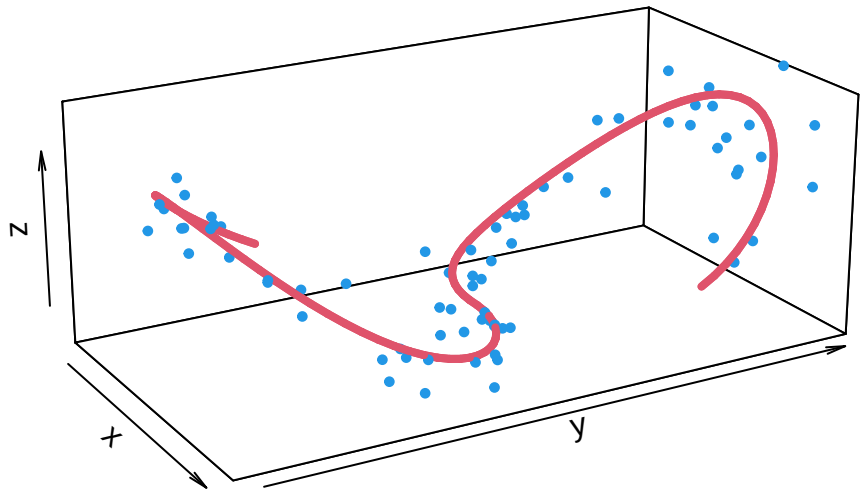
Principal curves, local MDS, Isomap and t-SNE

Caballero Vergés Biel, Menzenbach Svenja and Reyes Illescas Kleber Enrique

2023-10-23

PART A. Principal Curves

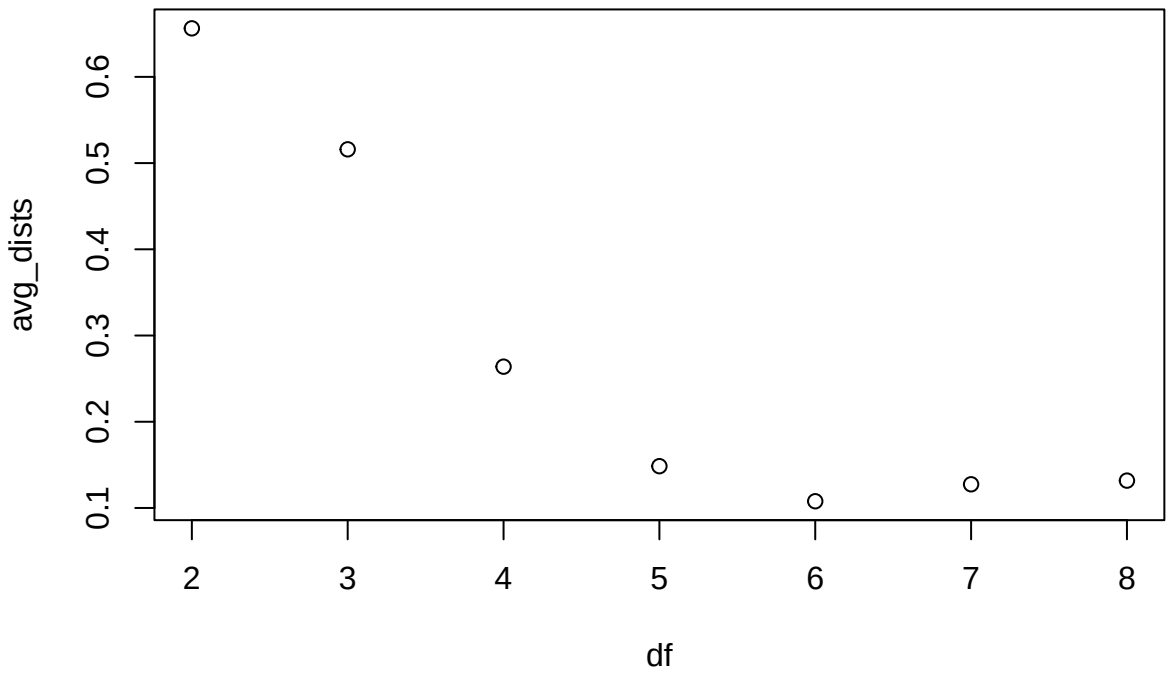
1.



Questions

a.

```r
library(princurve)

# function to determine the average sum-of-squared distances of loo principal curves by a given number
loo_proj_avg_dist <- function(X, df){
  N <- dim(X)[1]
  sum_dist <- 0
  for (n in 1:N){
    X_copy <- X
    fit_in <- principal_curve(X_copy[-n,], df=df)
    loo.proj <- project_to_curve(matrix(X[n,],ncol=3,byrow=TRUE), fit_in$s[fit_in$ord,])
    sum_dist <- sum_dist + loo.proj$dist
  }
  return(sum_dist/N)

}


df <- seq(2,8, by=1)

X <- cbind(rx, ry, rz)  # TODO: #cbind(rx, ry, rz) or cbind(x, y, z)?


# calculate avg sum-of-squared distances of loo principal curves for each df
avg_dists <- array(0, dim=c(1,length(df)))

for(i in df){
  avg_dists[i-1] <- loo_proj_avg_dist(X, df=i)
}


# determine optimal df
df_opt_idx <- which.min(avg_dists)
df_opt <- df[df_opt_idx]

plot(df, avg_dists)
```




```
print(df_opt)
```

```
## [1] 6
```

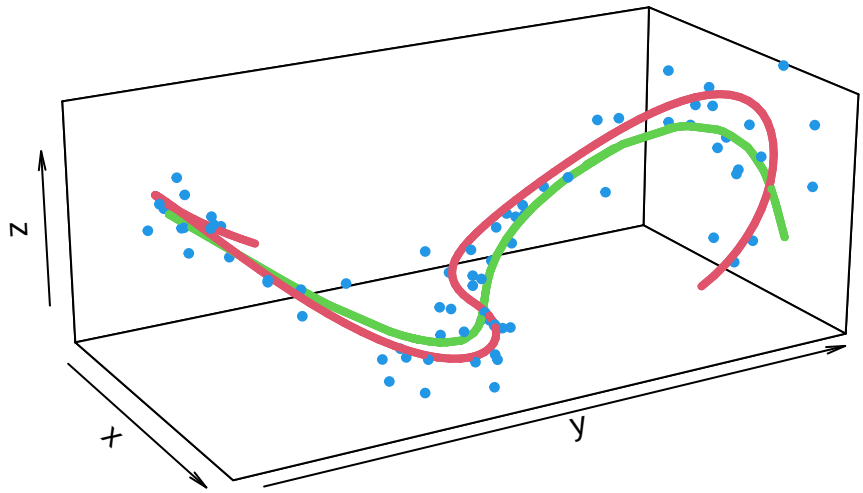
```
print(avg_dists[df_opt_idx])
```

```
## [1] 0.1077979
```

The optimal number of degree of freedom we obtain is 6 as this minimizes the average distance of the leave one out principal curves.

b.

```
# Give a graphical representation of the principal curve output for the optimal df and comment on the o

fit <- principal_curve(X, df=df_opt)
lines3D(fit$s[,1], fit$s[,2], fit$s[,3], colvar=NULL,
         phi = 20, theta = 60, r =sqrt(3), d =3, scale=FALSE,
         col=3,lwd=4,as=1,
         xlim=range(rx),ylim=range(ry),zlim=range(rz))
lines3D(x,y,z,colvar = NULL,
         phi = 20, theta = 60, r =sqrt(3), d =3, scale=FALSE,
         col=2,lwd=4,as=1,
         xlim=range(rx),ylim=range(ry),zlim=range(rz), add=TRUE)
points3D(rx,ry,rz,col=4,pch=19,cex=.6,add=TRUE)
```



The obtained principal curve (green) with df=6 is similar to the original curve. It represents the data quite well.

c.

```
avg_dist_50 <- loo_proj_avg_dist(X, df=50)
print(avg_dist_50)
```

```
## [1] 0.03939495
```

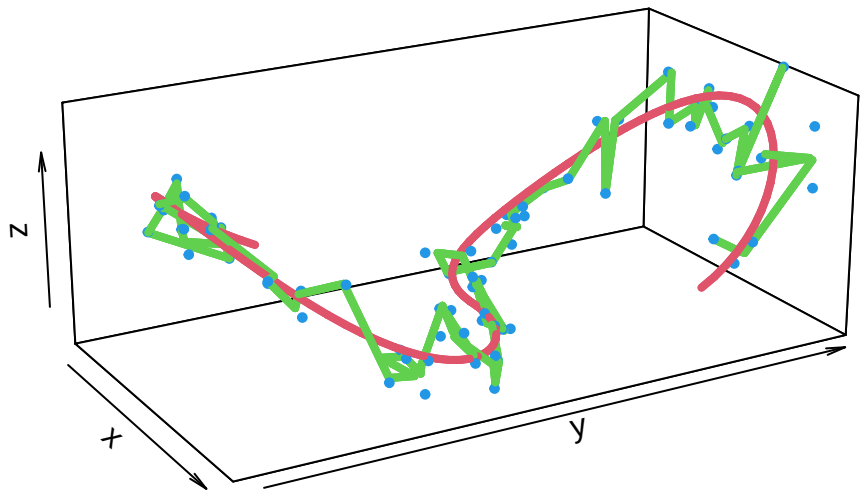
- Before fitting the principal curve with df=50 and based only on the leave-one-out cross-validation error values, what value for df do you think that is better, the previous optimal one or df=50?

Based on the leave-one-out cross-validation error values df=50 is better than the previous optimal one ($0.03939495 < 0.1077978$).

- Fit now the principal curve with df=50 and plot the fitted curve in the 3D scatterplot of the original points. Now, what value of df do you prefer?

```
fit <- principal_curve(X, df=50)
lines3D(fit$s[,1], fit$s[,2], fit$s[,3], colvar=NULL,
         phi = 20, theta = 60, r =sqrt(3), d =3, scale=FALSE,
         col=3,lwd=4,as=1,
         xlim=range(rx),ylim=range(ry),zlim=range(rz))
```

```
lines3D(x,y,z,colvar = NULL,
         phi = 20, theta = 60, r =sqrt(3), d =3, scale=FALSE,
         col=2,lwd=4,as=1,
         xlim=range(rx),ylim=range(ry),zlim=range(rz), add=TRUE)
points3D(rx,ry,rz,col=4,pch=19,cex=.6,add=TRUE)
```



I prefer the previous obtained optimal value here, because df=50 causes severe overfitting and hence does not generalize well to unseen data points.

- The overfitting with df=50 is clear. Nevertheless leave-one-out cross-validation has not been able to detect this fact. Why do you think that df=50 is given a so good value of leave-one-out cross-validation error?

The principal curve with df=50 is very close to the datapoints. So when projecting the points onto the curve the value is close to zero except for the left out one. However, this one higher distance is less than the sum of distances for df=6. So the curves seems better even though it just fits the data well that were used to obtain the principal curve.

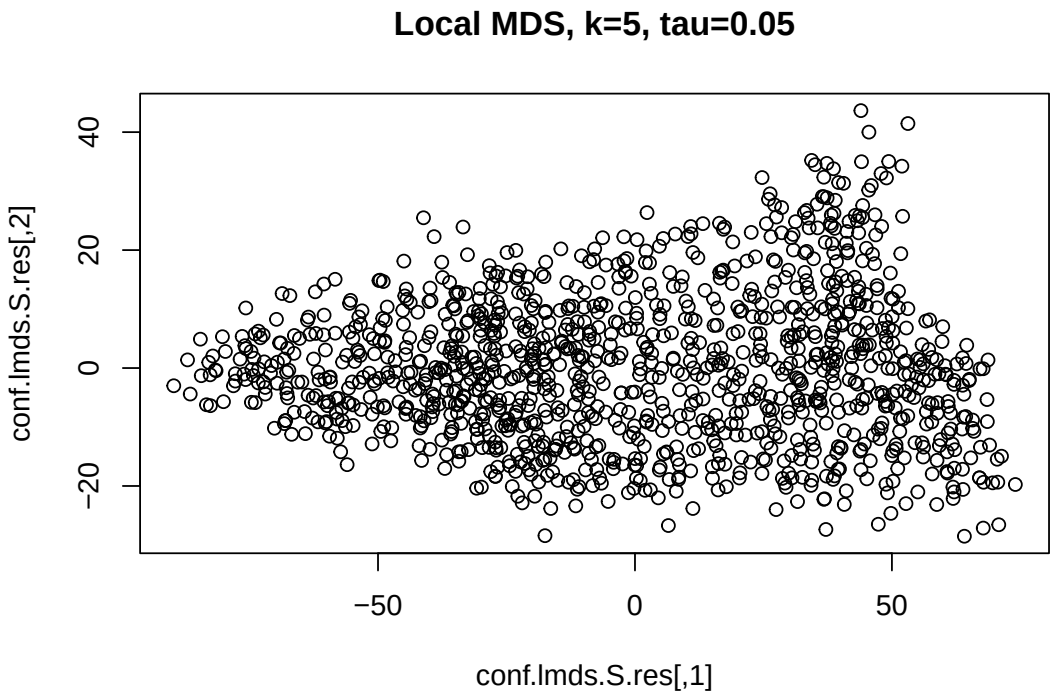
PART B. Local MDS, ISOMAP and t-SNE

2. Local MDS for ZERO digits

```
zip.train <- read.table("zip.train")
zip.train.0 <- zip.train[zip.train[1] == 0, -1]
```

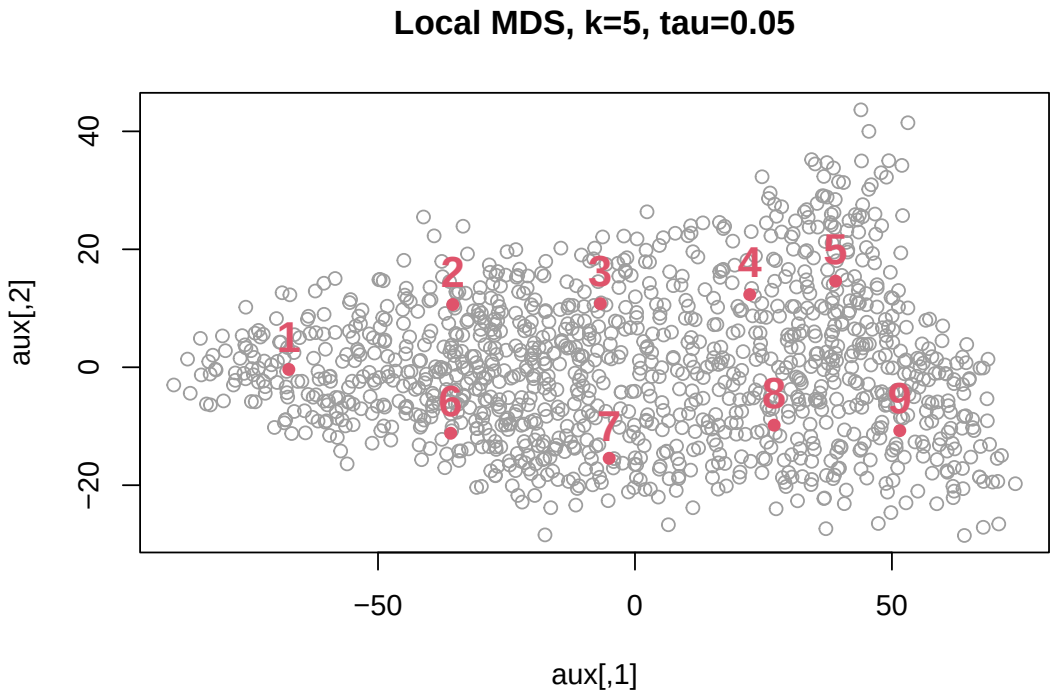
```
row.names(zip.train.0) <- NULL
dist.zip.0 <- dist(zip.train.0)

k <- 5
tau <- .05
q<-2 # 2-dim config

conf0 <- stats::cmdscale(dist.zip.0, k=q)

if (!file.exists("lmds1.RData")) {
  lmds.S.res <- lmds(as.matrix(dist.zip.0), init=conf0, ndim=q, k=k, tau=tau, itmax = 1000)
} else {
  load("lmds1.RData")
}

conf.lmds.S.res <- lmds.S.res$conf
```

```
plot(conf.lmds.S.res, main=paste0("Local MDS, k=",k,", tau=",tau))
```



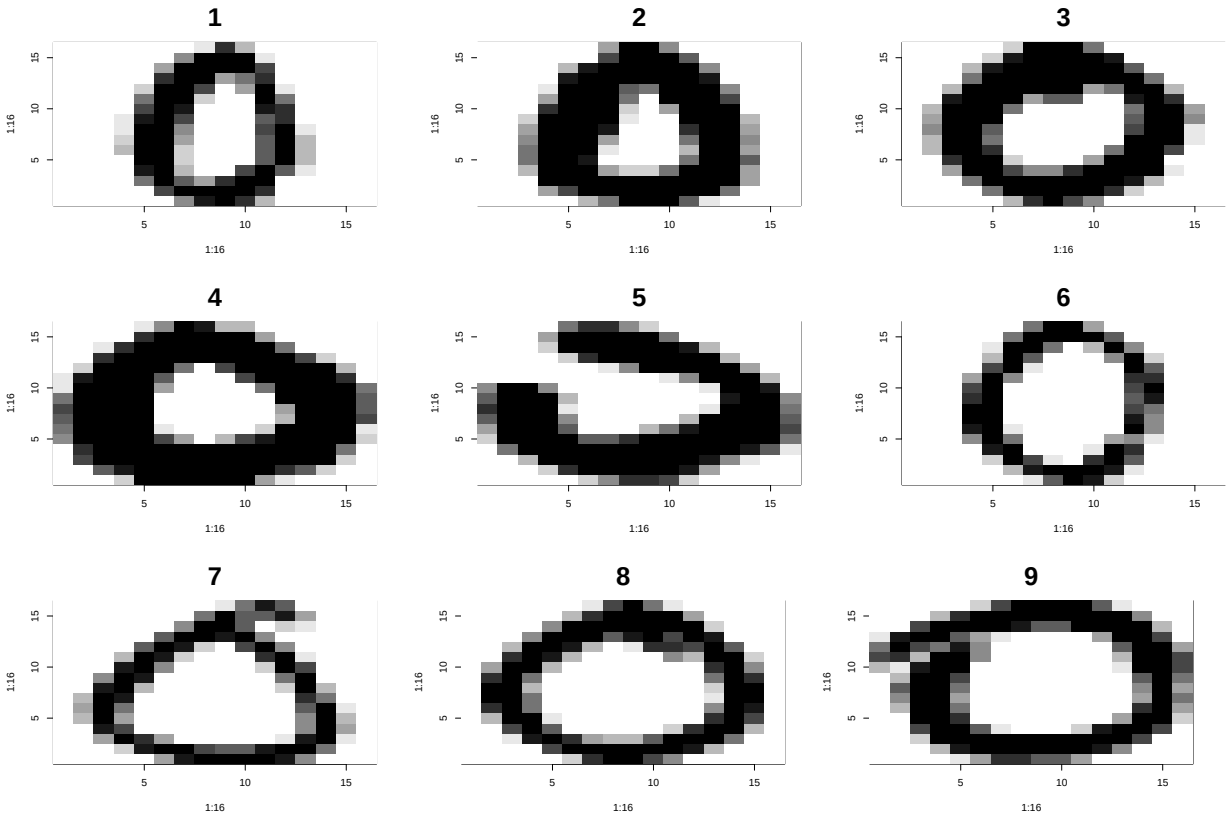
We've selected 9 points to encompass the data's variability. These points are marked in red, and each one has assigned a number in order to recognized each point with its representation using the *plot.zip* function.

```
#filt <- as.data.frame(conf.lmds.S.res)
#filt2 <- subset(filt, V1 > 40 & V1 < 55 & V2 < -7 & V2 > -14)
selected.points <- c(443, 575, 474, 291, 905, 731, 644, 242, 201)
aux <- conf.lmds.S.res[-selected.points,]
aux2 <- conf.lmds.S.res[selected.points,]

plot(aux, main=paste0("Local MDS, k=",k,", tau=",tau), col=8)
points(aux2, pch = 16, col=2)
text(aux2[,1],aux2[,2],1:9, pos=3, col=2, font = 2, cex = 1.5)
```



We can observe 2 patterns in the printed zeros below. The first one (the y-axis) is related to thickness of the stroke, positive values are thicker (the higher the value, the higher the stroke) and vice versa (lower values result in thinner strokes). On the other hand, the x-axis could be associated with the roundness of the number (negative values resemble an oval shape, while positive values resemble a circle shape). Furthermore, it appears that positive values in x-axis tend to result in unfinished strokes, like fast writing.

```
for (i in 1:length(selected.points)) {
  index <- selected.points[i]
  plot.zip(zip.train.0[index,], use.first=TRUE)
  title(i, cex.main = 2.5)
}
```



In this last step, we have used the LCMC in order to select the best parameters for k and τ. In this step we used a loop to calculate the overlapping between the high-dimension distances (**dist.zip.0**, previously calculated) and the low-dimension configuration distance obtained in each step. The optimal values are $k = 5$ and $\tau = 1$.

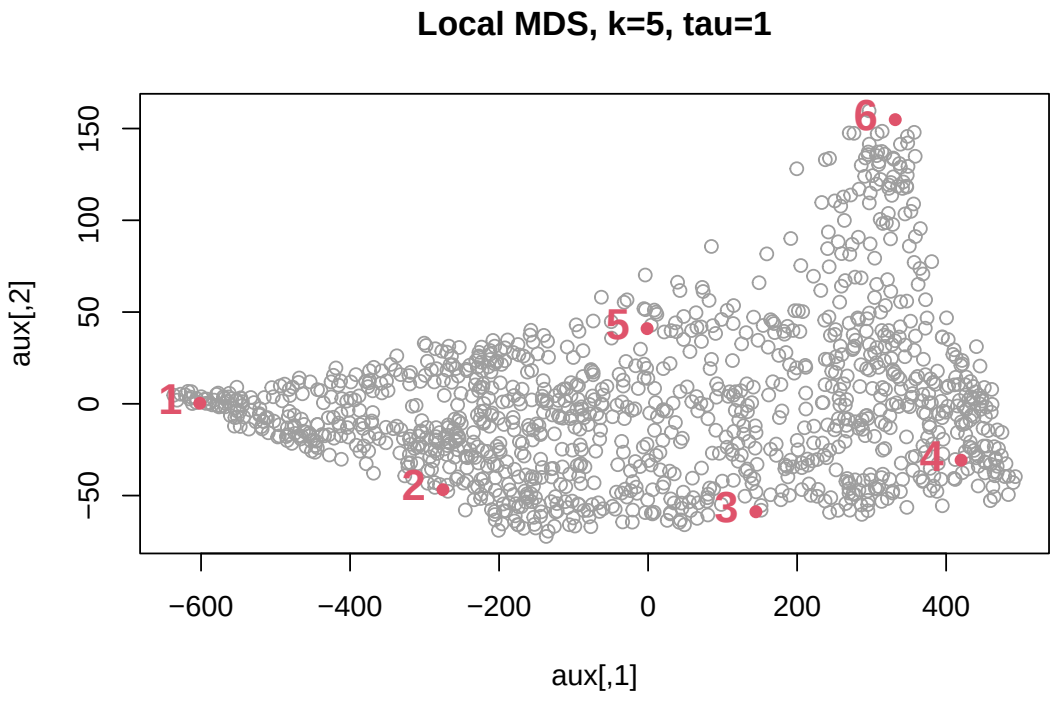
```
q <- 2
Kp <- 10

K <- c(5,10,15)
Tau <- c(.1,.5,1)

if (!file.exists("lmds_k_tau.RData")) {
  LC <- matrix(0,nrow=length(K),ncol=length(Tau))
  lmds.k.tau <- array(vector("list",1),dim=dim(LC))

  for (i in 1:length(K)){
    for (j in 1:length(Tau)){
      lmds.k.tau[[i,j]] <- lmds(as.matrix(dist.zip.0), init=conf0,
                                ndim=q, k=K[i], tau=Tau[j], itmax=1000)$conf
      D2.k.tau <- dist(lmds.k.tau[[i,j]])
      LC[i,j] <- LCMC(dist.zip.0,D2.k.tau,Kp)$M.Kp.adj
    }

  }
} else {
  load("lmds_k_tau.RData")
  load("LC.RData")
```

```
}


ij.max <- arrayInd(which.max(LC),.dim=dim(LC))
k.max <- K[ij.max[1]]
tau.max <- Tau[ij.max[2]]
lmds.max <- lmds.k.tau[[ij.max[1],ij.max[2]]]

print(paste0("k.max=",k.max,"; tau.max=",tau.max))
```

```
## [1] "k.max=5; tau.max=1"
```

Finally, we can once again observe the plot but with the new configuration. We selected new points to confirm the interpretations we provided earlier with the first configuration.

```
selected.points <- c(344, 962, 221, 530, 399, 440)
aux <- lmds.max[-selected.points,]
aux2 <- lmds.max[selected.points,]

plot(aux, main=paste0("Local MDS, k=",k.max,", tau=",tau.max), col=8)
points(aux2, pch = 16, col=2)
text(aux2[,1],aux2[,2],1:6, pos=2, col=2, font = 2, cex = 1.5)
```

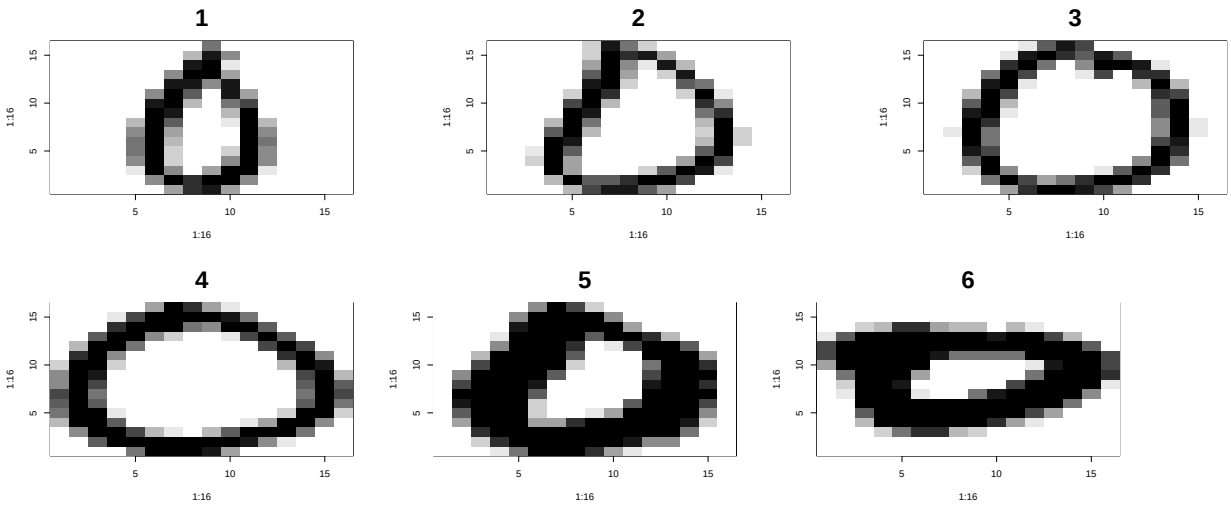


In this case, we can confirm that y-axis is correlated with the thickness of the number (figures 5 and 6 are thicker while the others are thinner). and x-axis is correlated with the roundness.

It's interenting to notice that in the left side of the points (points around 1), there is not much variability in the y-axis, while on the right side, the variability is the greatest. This give us a sort of a triangular form

that can be related with our initial observation about writing velocity (fast writing leads to more variations of the symbol compared to the standard form of the number zero).

```
for (i in 1:length(selected.points)) {
  index <- selected.points[i]
  plot.zip(zip.train.0[index,], use.first=TRUE)
  title(i, cex.main=2.5)
}
```



3.

4.

5.