Non-linear dimensionality reduction

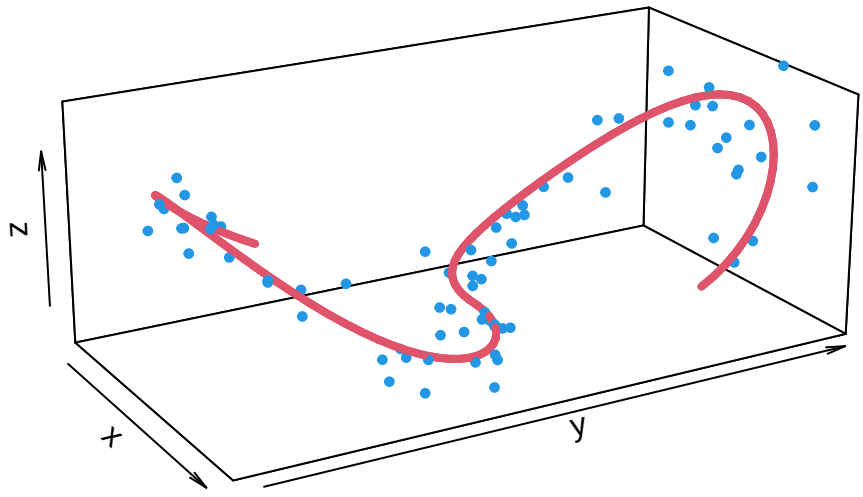
Principal curves, local MDS, Isomap and t-SNE

Caballero Vergés Biel, Menzenbach Svenja and Reyes Illescas Kleber Enrique

2023-10-25

PART A. Principal Curves

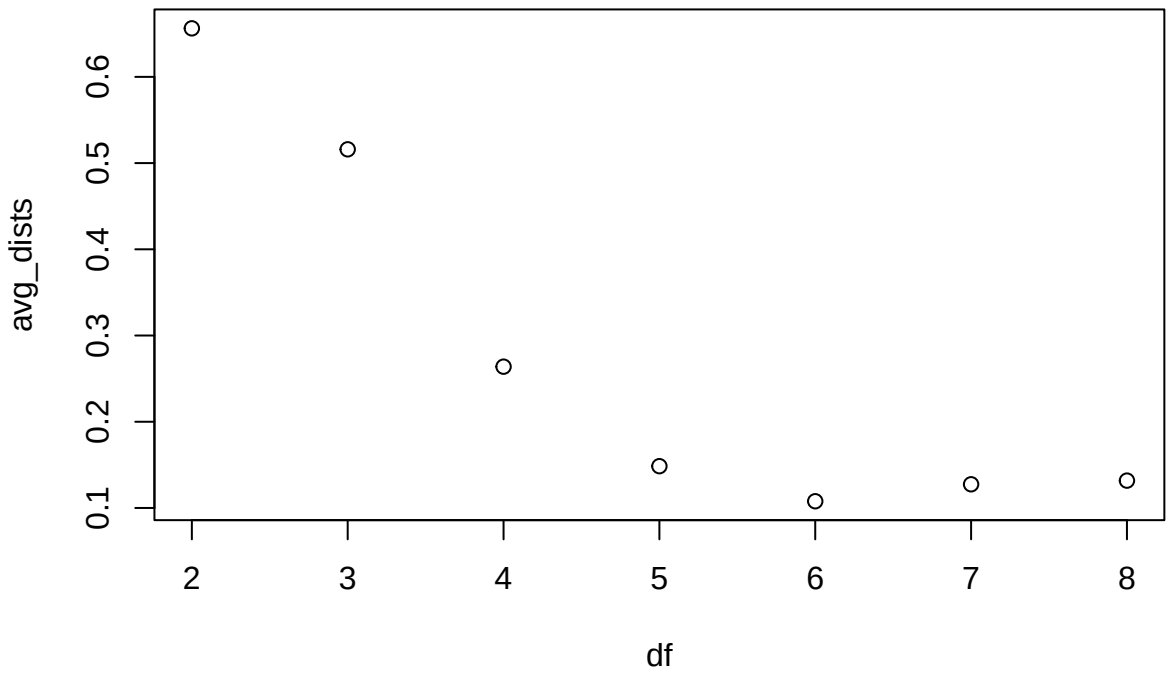
1.



Questions

a.

```r
# function to determine the average sum-of-squared distances
# of loo principal curves by a given number of df
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)


# 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
```

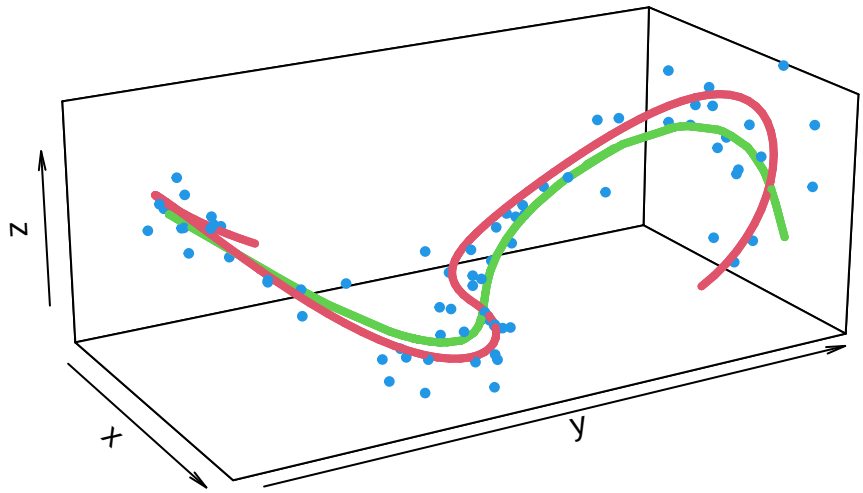
```
print(avg_dists[df_opt_idx])
```

```
## [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.

```
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
```

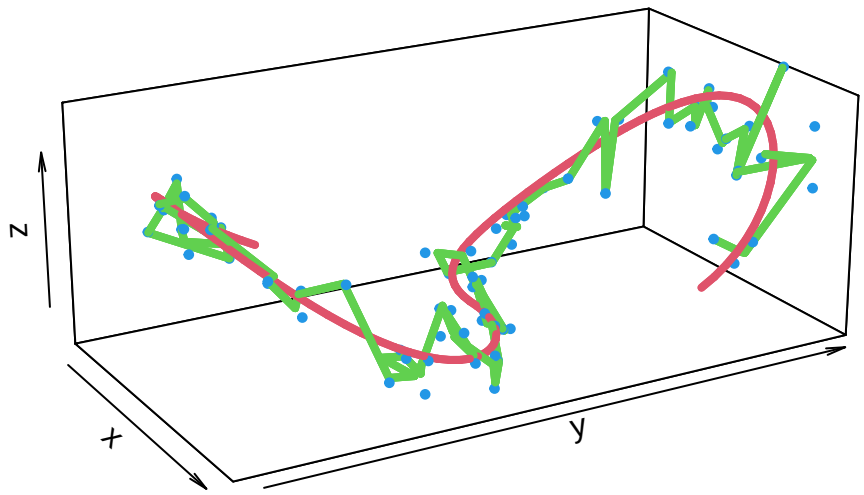
- Before fitting the principal curve with df=50 and based only on the leave-one-out cross-validation error values, what value for df do you think that is better, the previous optimal one or df=50?

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.

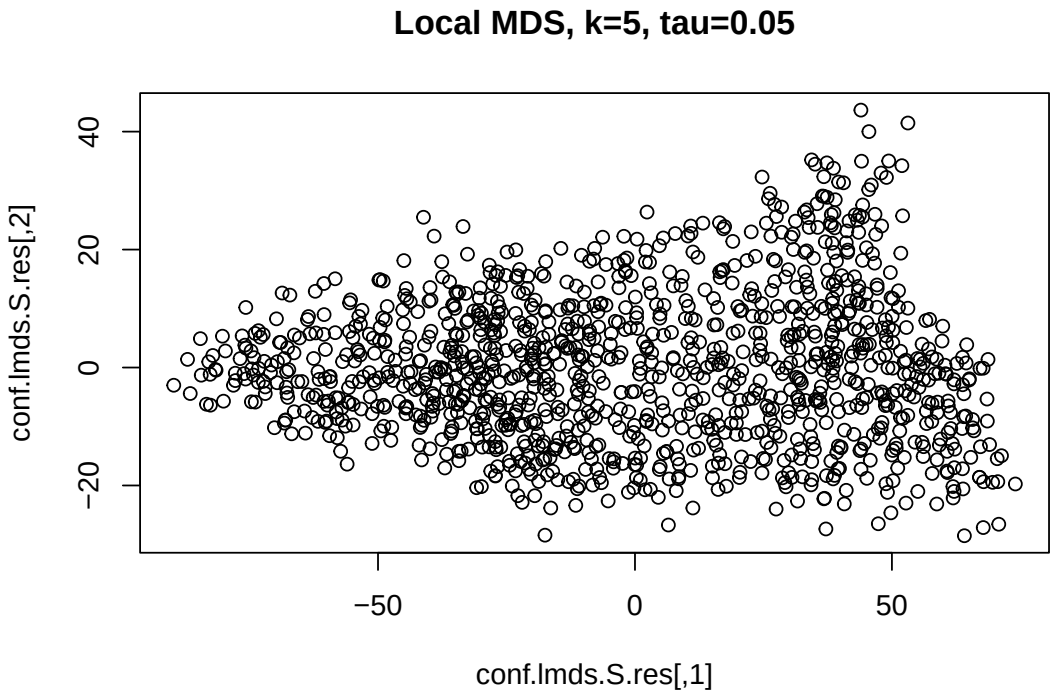
PART B. Local MDS, ISOMAP and t-SNE

2. Local MDS for ZERO digits

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

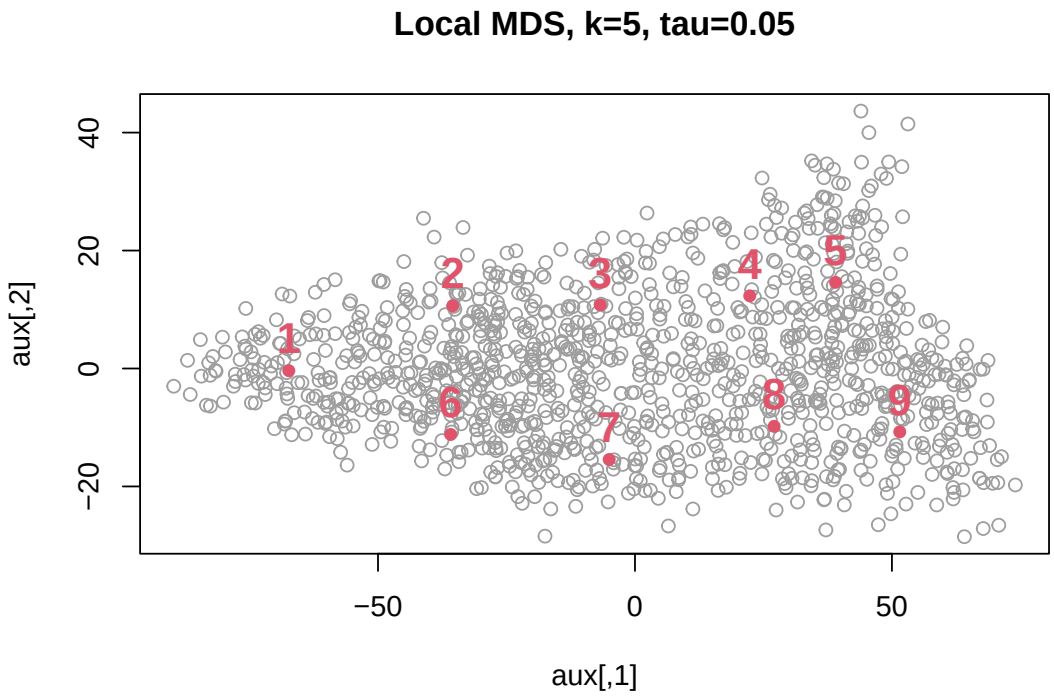
```
row.names(zip.train.0) <- NULL
dist.zip.0 <- dist(zip.train.0)

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))
```



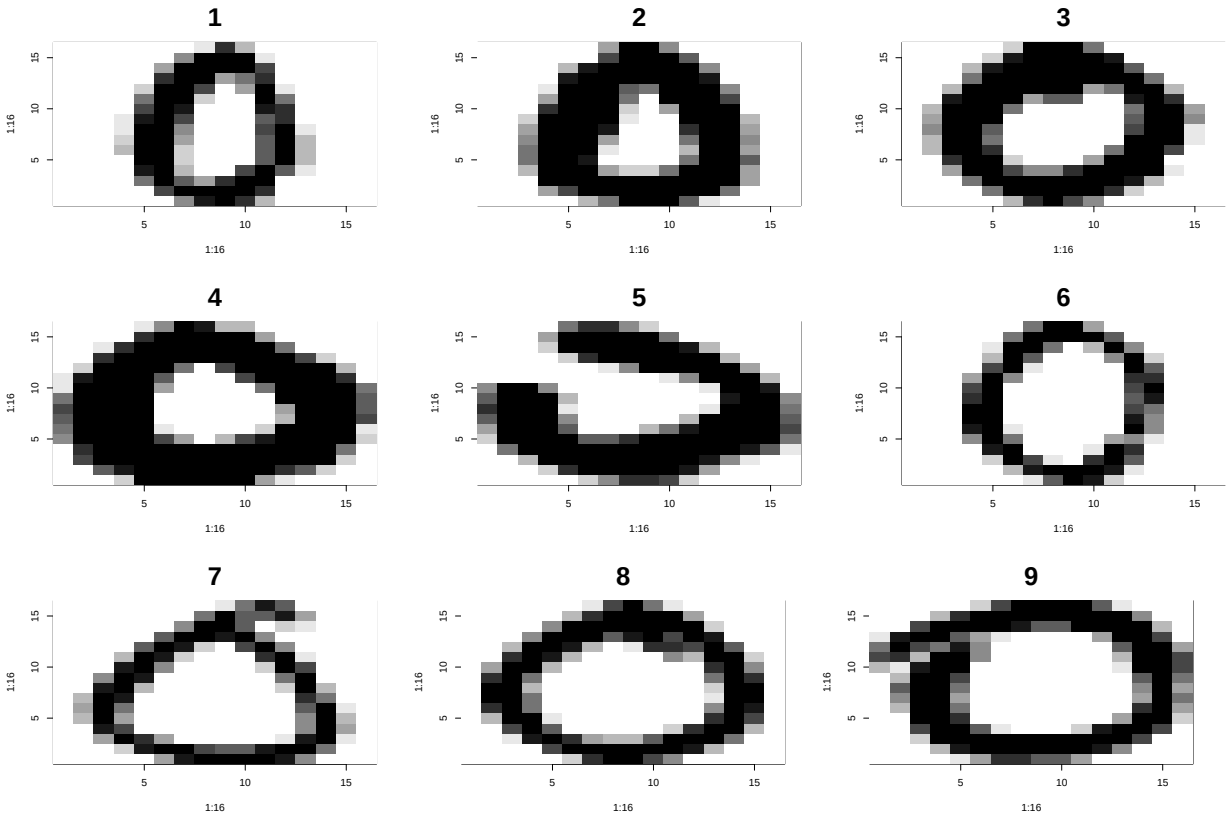
We've selected 9 points to encompass the data's variability. These points are marked in red, and each one has assigned a number in order to recognized each point with its representation using the *plot.zip* function.

```
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 narrowness 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 $\tau = 1$.

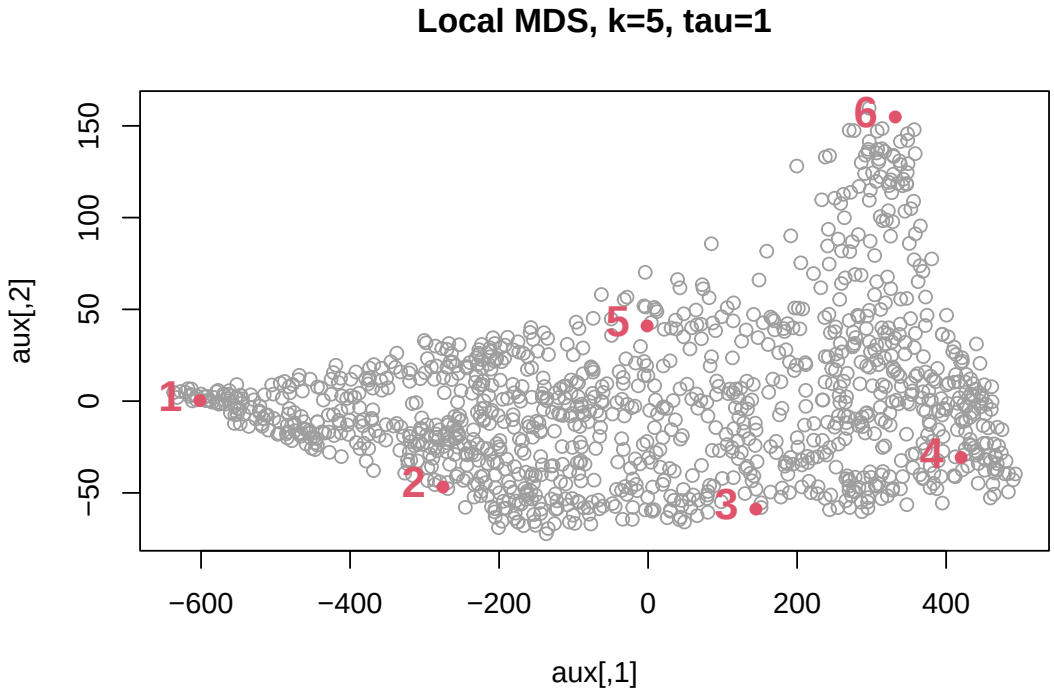
```
q <- 2
Kp <- 10

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,
             "; LC.max=", LC[ij.max[1], ij.max[2]]))
```

```
## [1] "k.max=5; tau.max=1; LC.max=0.2628070500589"
```

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.

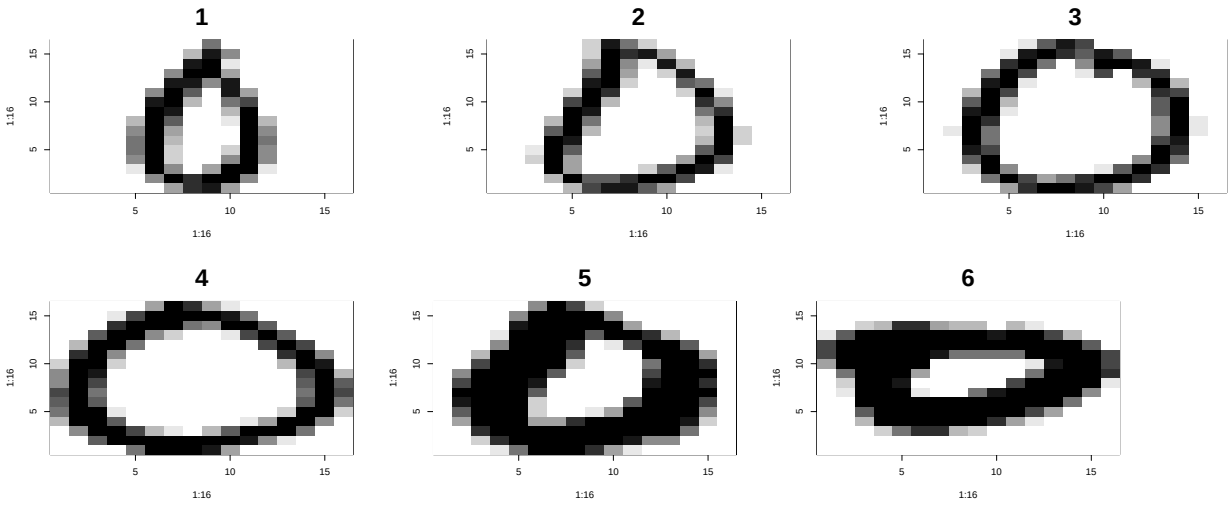
```
selected.points <- c(344, 962, 221, 530, 399, 440)
aux <- lmds.max[-selected.points,]
aux2 <- lmds.max[selected.points,]

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 interesting 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).

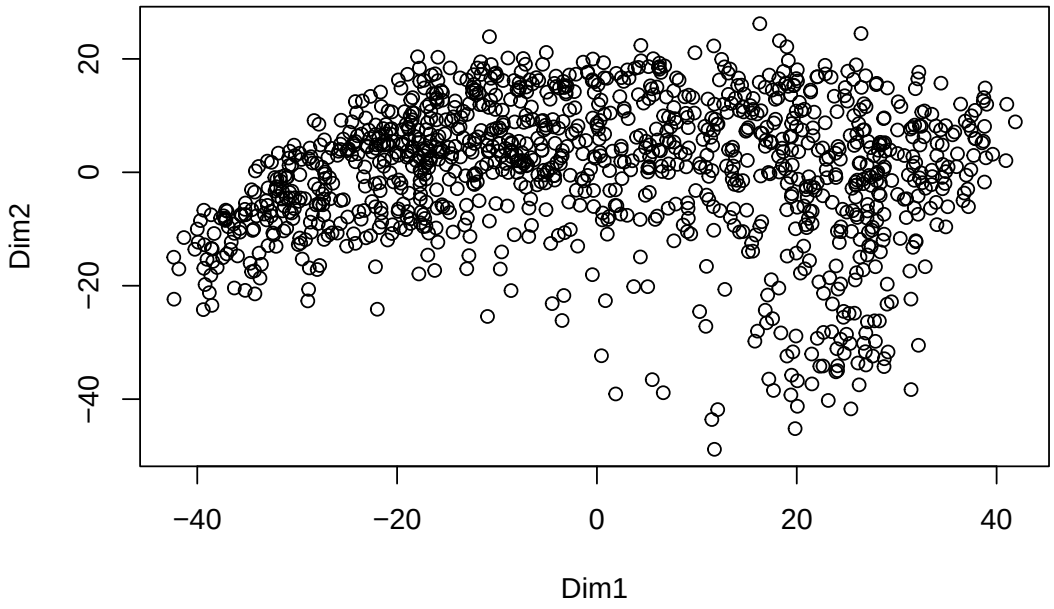
```
for (i in 1:length(selected.points)) {
  index <- selected.points[i]
  plot.zip(zip.train.0[index,], use.first=TRUE)
  title(i, cex.main=2.5)
}
```



3. ISOMAP for ZERO digits

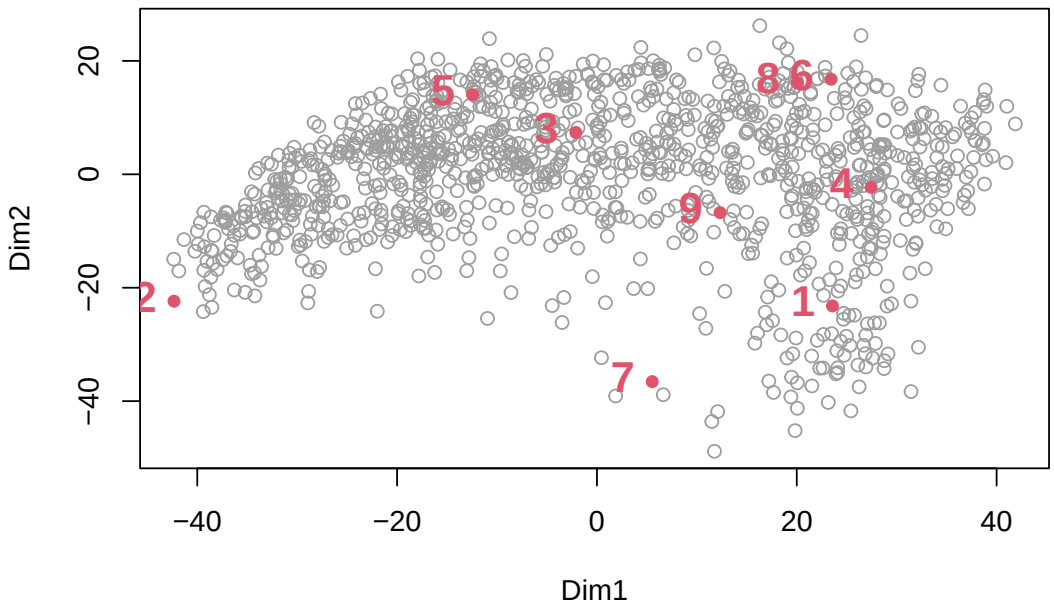
```
library(vegan)
row.names(zip.train.0) <- NULL
dist.zip.0 <- dist(zip.train.0)

k <- 5
if (!file.exists("isomap_out.RData")) {
  isomap_out <- isomap(dist.zip.0, ndim = 2, k = k)
} else {
  load("isomap_out.RData")
}
```

```
plot(isomap_out$points)
```

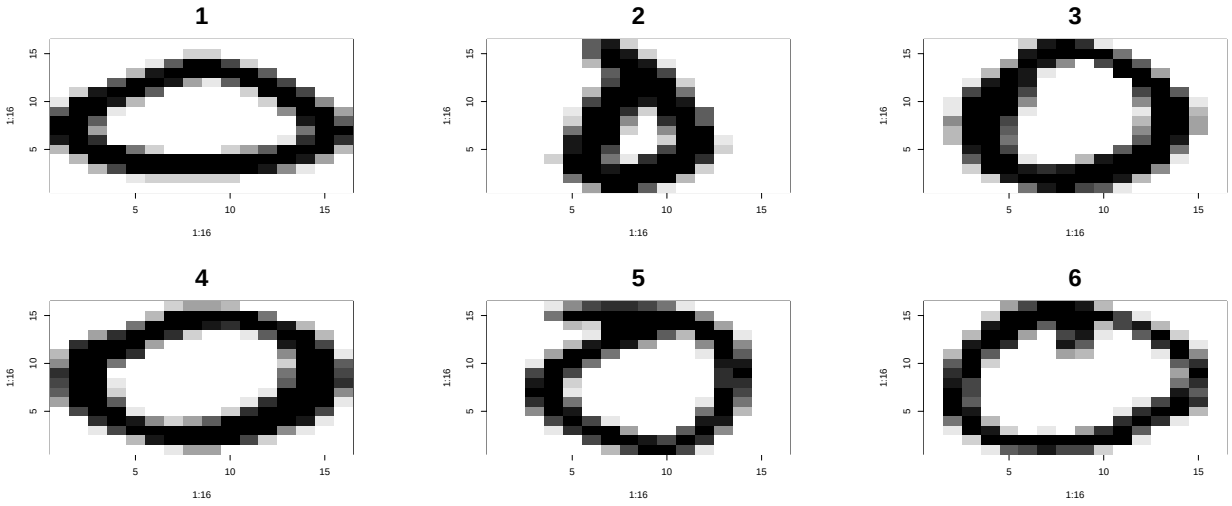


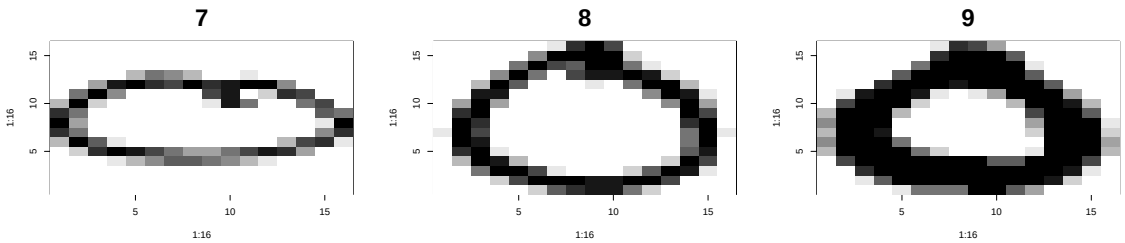
In this case Isomap gives us a peculiar result since x-axis still relates to narrowness as before, but we cannot say that y-axis and thickness are related. Instead, we observe a sort of gaussian distribution around the point 9 (being a local maximum). As we move further away from this point, the numbers become thinner. Given that 7 is evidently the thinnest point in our example (in contrast to 2, for instance), we may suspect that there exist a hidden dimension that is not well represented. In this case we have a 2-dimension representation of a 3-dimension surface, so the reduction didn't complete correctly.

```
selected.points <- c(318, 1078, 127, 36, 514, 165, 1131, 486, 650)
isomap_out.no.selected <- isomap_out$points[-selected.points,]
isomap_out.selected <- isomap_out$points[selected.points,]

plot(isomap_out.no.selected, col=8)
points(isomap_out.selected, pch = 16, col=2)
text(isomap_out.selected[,1],isomap_out.selected[,2],1:9,
     pos=2, col=2, font = 2, cex = 1.5)
```

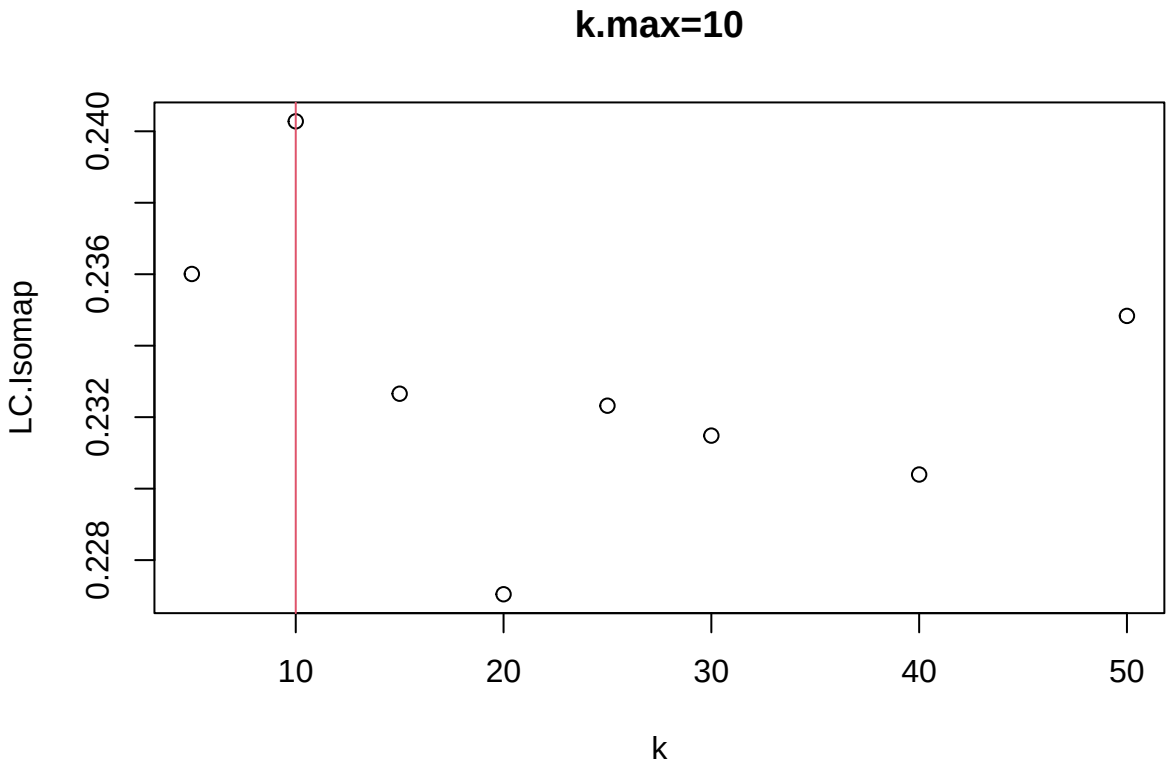



```
for (i in 1:length(selected.points)) {
  index <- selected.points[i]
  plot.zip(zip.train.0[index,], use.first=TRUE)
  title(i, cex.main=2.5)
}
```






```
Kp <- 10
k <- c(5,10,15,20,25,30,40,50)

if (!file.exists("Isomap_k.RData")) {
  LC.Isomap <- numeric(length(k))
  Isomap.k <- vector("list",length(k))
  for (i in 1:length(k)){
    Isomap.k[[i]] <- isomap(dist.zip.0, ndim = 2, k = k[i])
    D2.k <- dist(Isomap.k[[i]]$points)
    LC.Isomap[i] <- LCMC(dist.zip.0,D2.k,Kp)$M.Kp.adj
  }
} else {
  load("LC_isomap.RData")
  load("isomap_k.RData")
}

i.isomap.max <- which.max(LC.Isomap)
k.max <- k[i.isomap.max[1]]
Isomap.max <- Isomap.k[[i.isomap.max]]

plot(k, LC.Isomap, main=paste0("k.max=",k.max))
abline(v=k[i.isomap.max],col=2)
```

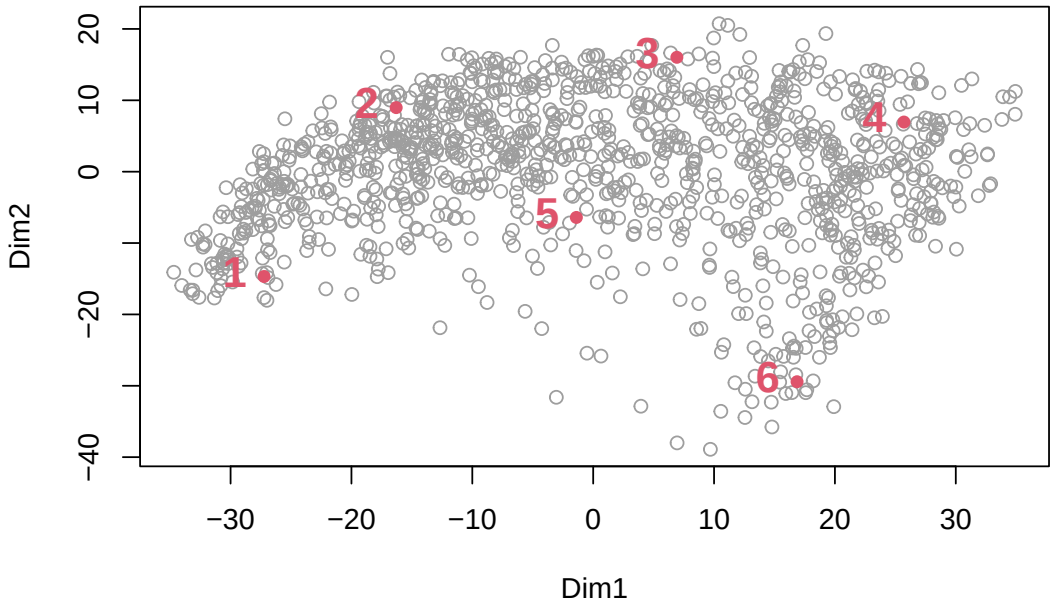


Now, we have a more similar distribution than before. With this new parameter we have reached to a configuration that can be divided into 2 dimension. In this case, y-axis is not perfectly correlated with the thickness, as both values 1 and 6 have a similar value for the y-axis. This suggest that now thickness is also correlated with the x-axis and we require both dimensions to explain the thickness of a point.

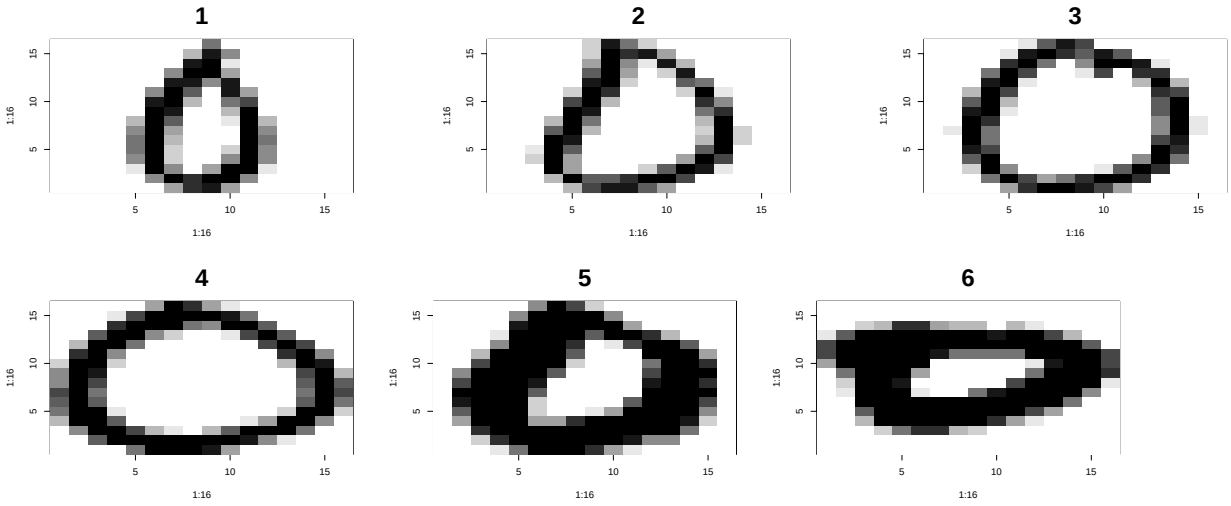
```
selected.points <- c(344, 962, 221, 530, 399, 440)
aux <- Isomap.max$points[-selected.points,]
aux2 <- Isomap.max$points[selected.points,]

plot(aux, main=cat("Isomap, k=",k.max), col=8)
```

```
## Isomap, k= 10
```

```
points(aux2, pch = 16, col=2)
text(aux2[,1],aux2[,2],1:6, pos=2, col=2, font = 2, cex = 1.5)
```



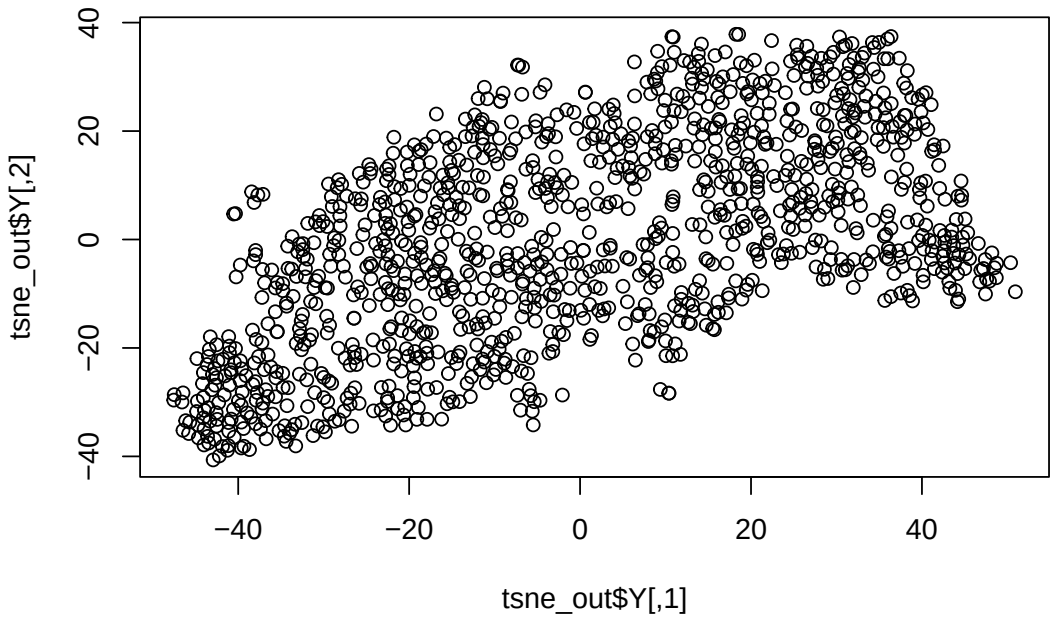

```
for (i in 1:length(selected.points)) {
  index <- selected.points[i]
  plot.zip(zip.train.0[index,], use.first=TRUE)
  title(i, cex.main=2.5)
}
```



4. t-SNE for ZERO digits

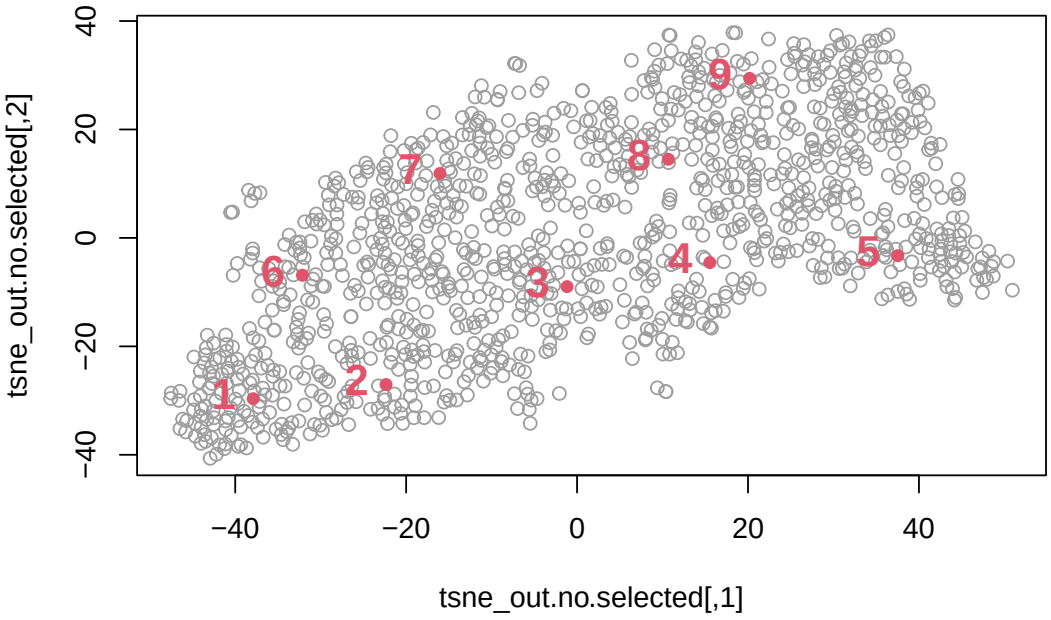
```
#row.names(zip.train.0) <- NULL
#zip_matrix <- as.matrix(zip.train.0)

if (!file.exists("tsne_out.RData")) {
  tsne_out <- Rtsne(dist.zip.0, dims=2, pca=FALSE,
                    perplexity=40,theta=0.0, max_iter=1000,
                    num_threads=4)
} else {
  load("tsne_out.RData")
}
```

```
plot(tsne_out$Y)
```

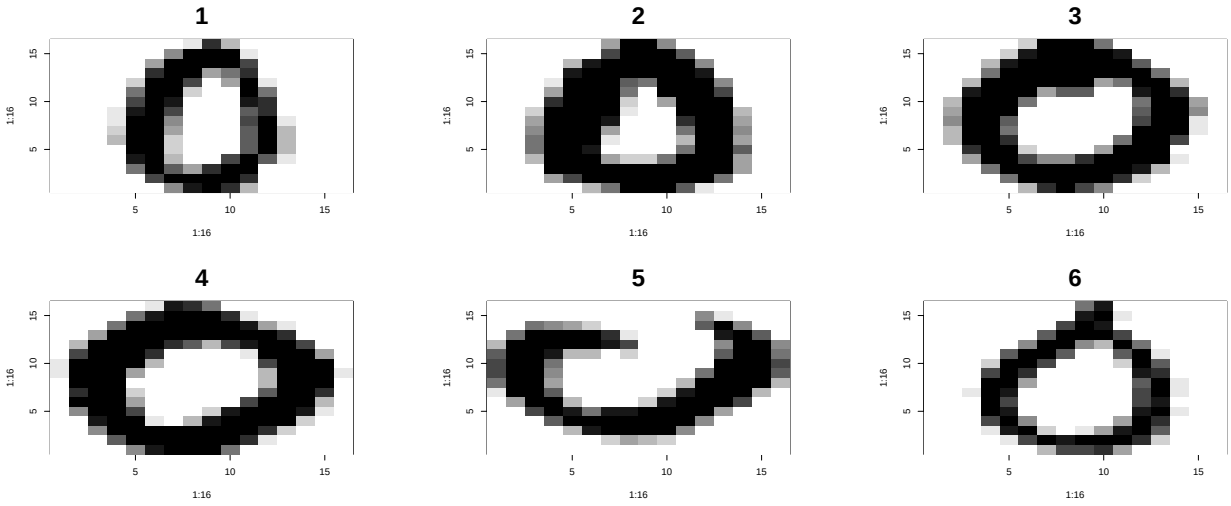


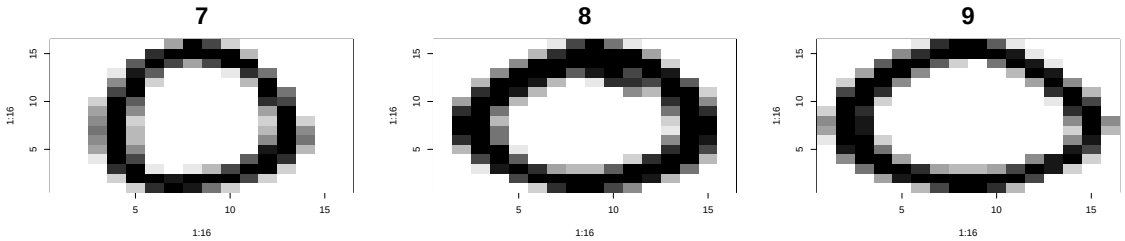
In this case, we can notice that the y-axis tends to increase as the x-axis increase, but the conclusions are similar to previous exercises. The y-axis is correlated with thickness and x-axis with narrowness It's worth highlighting that in this case, lower values correspond to thickness, but the overall interpretation remains more or less the same. Something that differs is that the thickness is not explained only by the y-axis since values like 9 and 6 have the same degree of thickness.

```
selected.points <- c(443, 575, 474, 139, 489, 732, 198, 242, 699 )
tsne_out.no.selected <- tsne_out$Y[-selected.points,]
tsne_out.selected <- tsne_out$Y[selected.points,]

plot(tsne_out.no.selected, col=8)
points(tsne_out.selected, pch = 16, col=2)
text(tsne_out.selected[,1],tsne_out.selected[,2],1:9, pos=2, col=2, font = 2, cex = 1.5)
```



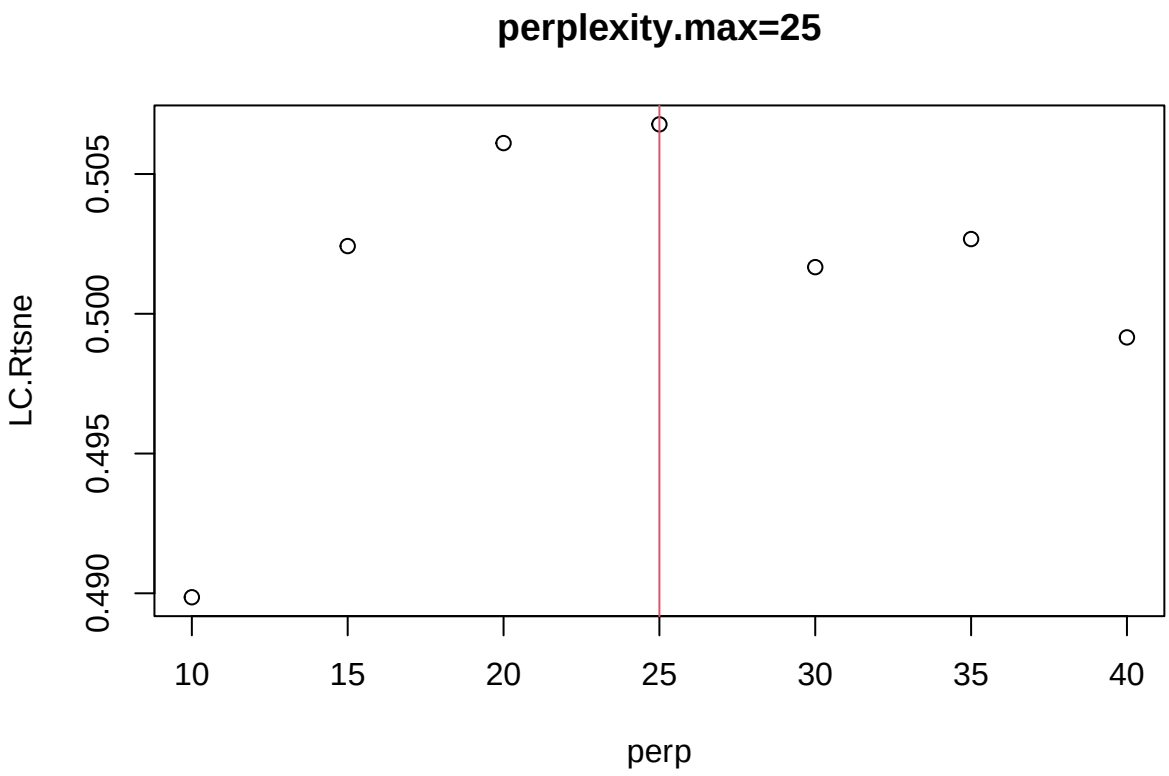

```
for (i in 1:length(selected.points)) {
  index <- selected.points[i]
  plot.zip(zip.train.0[index,], use.first=TRUE)
  title(i, cex.main=2.5)
}
```





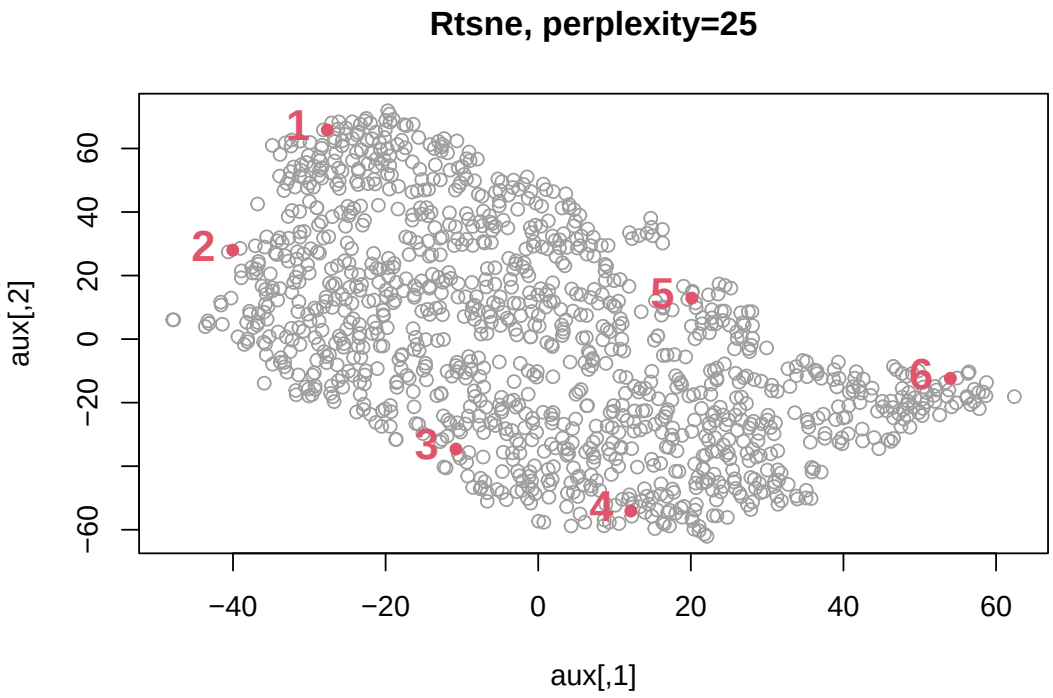
LCMC gives us as a result that the best value for perplexity is 25.

```
Kp <- 10
perp <- c(10,15,20,25,30,35,40)

if (!file.exists("Rtsne_perp.RData")) {
  LC.Rtsne <- numeric(length(perp))
  Rtsne.perp <- vector("list",length(perp))

  for (i in 1:length(perp)){
    Rtsne.perp[[i]] <- Rtsne(dist.zip.0, dims=2, pca=FALSE,
                             perplexity=perp[i],theta=0.0, max_iter=1000,
                             num_threads=4)
    D2.perp <- dist(Rtsne.perp[[i]]$Y)
    LC.Rtsne[i] <- LCMC(dist.zip.0,D2.perp,Kp)$M.Kp.adj

  }
} else {
  load("LC_rtsne.RData")
  load("Rtsne_perp.RData")
}

i.rtsne.max <- which.max(LC.Rtsne)
perplexity.max <- perp[i.rtsne.max[1]]
Rtsne.max <- Rtsne.perp[[i.rtsne.max]]

plot(perp, LC.Rtsne, main=paste0("perplexity.max=",perplexity.max))
abline(v=perp[i.rtsne.max],col=2)
```

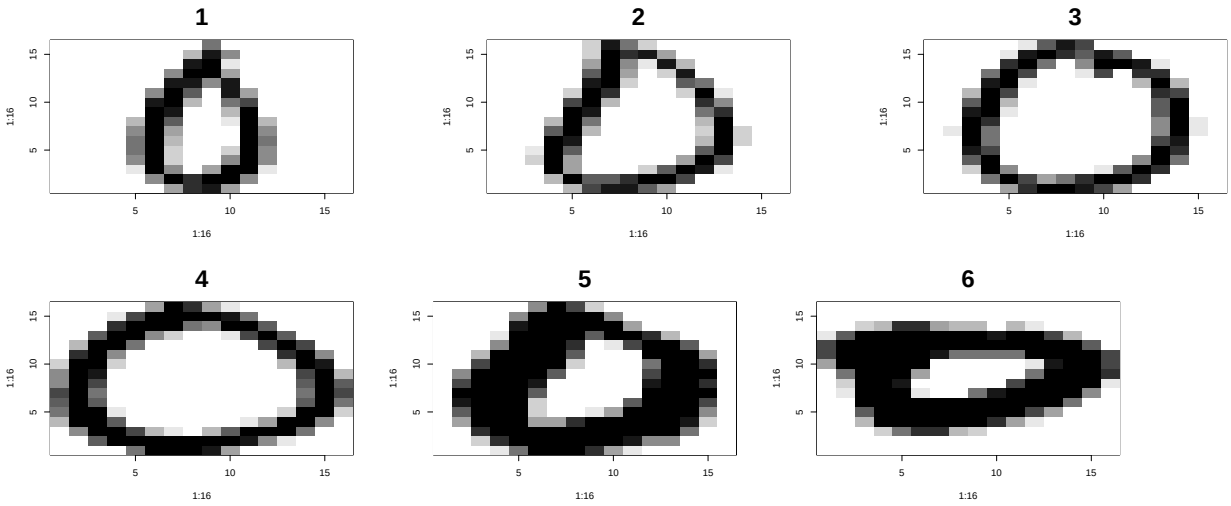


In this part, we used the same points as in B2c part. The conclusions are similars, but we must note that thickness is not perfectly represented by y-axis as in the previous examples. Instead, we can draw an imaginary line between points 5,6 and 1,2,3,4, which separates the level of thickness. The left side represented by points 1,2,3,4 are thinner than the group on the right side.

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

plot(aux, main=paste0("Rtsne, perplexity=",perplexity.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)
```




```
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)
}
```



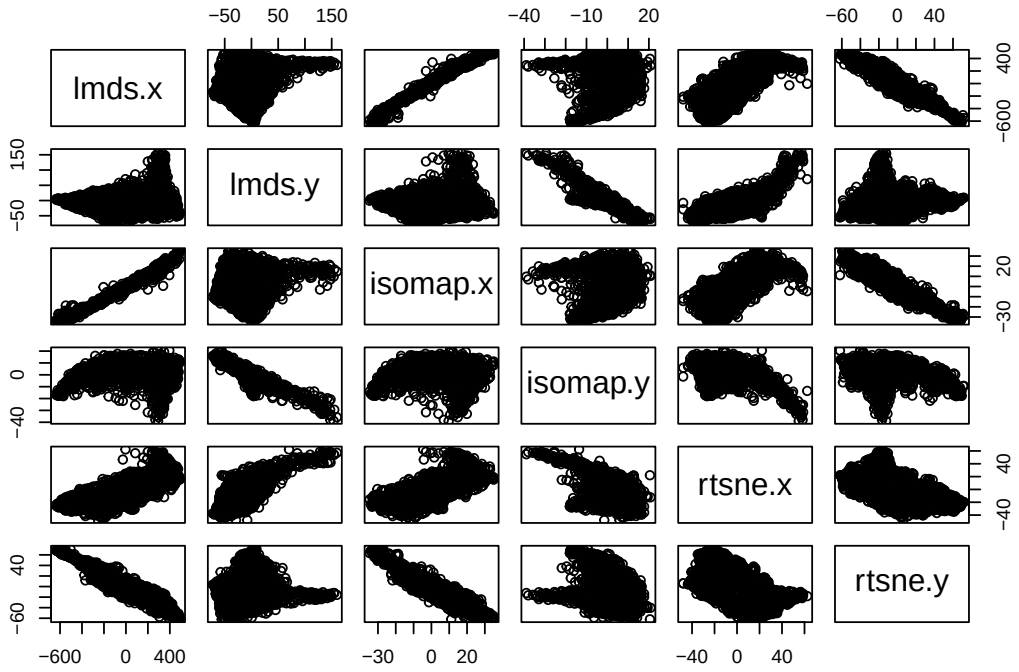
5. Compare Local MDS, ISOMAP and t-SNE for ZERO digits

In the plot above, we can compare the 2-dimensions of the 3 configurations. First, looking at the x-axis, we can notice that lmds and isomap are clearly positive correlated. However, when we compare both of them

with t-sne we can realize how the variation increases. The correlation remains positive, but now points are sparser than before (Remember that x-axis is related to the narrowness of the number).

On the other hand, when we compare the y-axis, we can confirm a negative correlation between lmds and isomap (this is due to thickness values in lmds are on the top while thickness values in isomap are in the bottom), but in the case of t-sne, the relationship is not clear at all. We must remember that thickness is not completely correlated with the y-axis and that's why the distributions shown here have these strange shapes.

```
pairs(cbind(lmds=data.frame("x" = lmds.max[,1],"y" = lmds.max[,2]),
            isomap=data.frame("x"= Isomap.max$points[,1],"y"= Isomap.max$points[,2]),
            rtsne=data.frame("x"= Rtsne.max$Y[,1],"y"= Rtsne.max$Y[,2])))
```



We can also compare the LCMC values for the 3 methods. The t-sne reached the maximum at 0.51 (with perplexity=25). This can help us decide which is better In this case, t-sne seems the obvious options. Furthermore, we must mention that it takes less time for computation, which is always a plus.

```
data.frame(LMDS=LC[ij.max[1], ij.max[2]],
           Isomap=LC.Isomap[i.isomap.max],
           TSNE=LC.Rtsne[i.rtsne.max], row.names = c("LC"))
```

```
##         LMDS    Isomap      TSNE
## LC 0.2628071 0.2402777 0.5067769
```

In order to improve the results or to confirm the conclusions, we can repeat the tuning process with more values for the perplexity. It's also important to mention that even though the Meta-criteria score of LMDS is lower its distributions is easier to explain.