Dimension Reduction, AMSI 2021 Winter School

Tutorial 3 and 4 Solutions

Anastasios Panagiotelis

July 15, 2021

Images Data - Cleaning

1. Using the images data construct a matrix where each row corresponds to an image and each column to a color channel (R, G or B) of an individual pixel.

This answer assumes you have downloaded the images data locally (by cloning the github repo. Alternatively you can download the data directly from online by replacing the local file location with a url.

```
library(tidyverse)
library(png)
library(dimRed)
filenames<-list.files('../../data/images/')
# First store everything in a matrix
imagedat<-matrix(0,length(filenames), 124848) #Initialise matrix
for(i in 1:length(filenames)){
  out<-readPNG(paste0('../../data/images/',filenames[i]))%>%as.vector()
  imagedat[i,]<-out
}
#Make column names easier to read
colnames(imagedat)<-paste0('V',str_pad(1:124848,6,'left','0'))
```

2. Find all variables which have no variation across the images and remove them.

```
vars<-apply(imagedat,2,var)
imagedat<-imagedat[,(vars>0)]
```

3. Convert the matrix to a data frame and add a column to your dataframe with the name of each file.

```
pix<-as_tibble(imagedat)
pix<-add_column(pix,Image=filenames,.before = 1)
head(pix)
```

```
## # A tibble: 6 x 10,898
##   Image  V016623 V016818 V016820 V016821 V016822 V016825 V016827 V016828 V017020
##   <chr>    <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>
## 1 A0001~       1       1       1       1       1       1       1       1       1
## 2 A0002~       1       1       1       1       1       1       1       1       1
## 3 A0003~       1       1       1       1       1       1       1       1       1
## 4 A0004~       1       1       1       1       1       1       1       1       1
## 5 A0005~       1       1       1       1       1       1       1       1       1
## 6 A0006~       1       1       1       1       1       1       1       1       1
## # ... with 10,888 more variables: V017021 <dbl>, V017022 <dbl>, V017023 <dbl>,
## #   V017024 <dbl>, V017025 <dbl>, V017026 <dbl>, V017029 <dbl>, V017030 <dbl>,
## #   V017031 <dbl>, V017032 <dbl>, V017033 <dbl>, V017219 <dbl>, V017220 <dbl>,
## #   V017224 <dbl>, V017225 <dbl>, V017226 <dbl>, V017227 <dbl>, V017228 <dbl>,
## #   V017229 <dbl>, V017230 <dbl>, V017231 <dbl>, V017232 <dbl>, V017233 <dbl>,
## #   V017234 <dbl>, V017235 <dbl>, V017236 <dbl>, V017237 <dbl>, V017238 <dbl>,
## #   V017240 <dbl>, V017241 <dbl>, V017242 <dbl>, V017243 <dbl>, ...
```

Images Data - Dimension Reduction

1. Carry out PCA and ISOMAP using the `dimRed` package (note some commands may take about 20-30 second to run - be patient!). Use the function defaults for output dimension and tuning parameters.

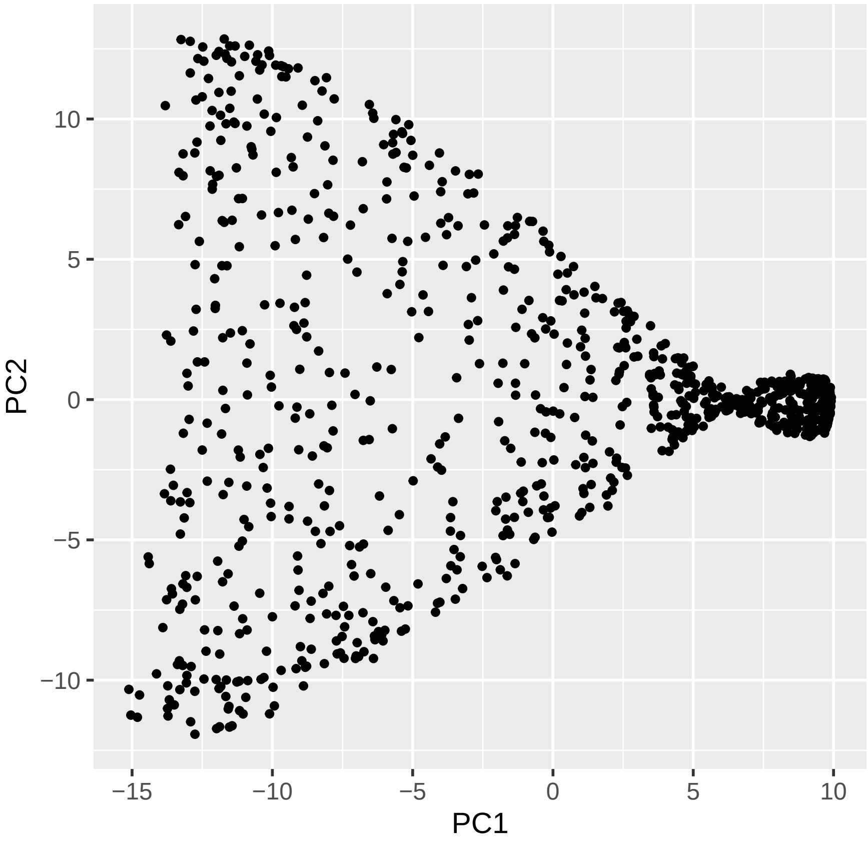
```
#Get into the necessary S4 class
im<-as.dimRedData(Image~.,data=pix)
pcaout<-embed(.data = im, .method="PCA")
isoout<-embed(.data = im, .method="Isomap")
```

2. For PCA and isomap, plot a scatterplot.

Plot for PCA

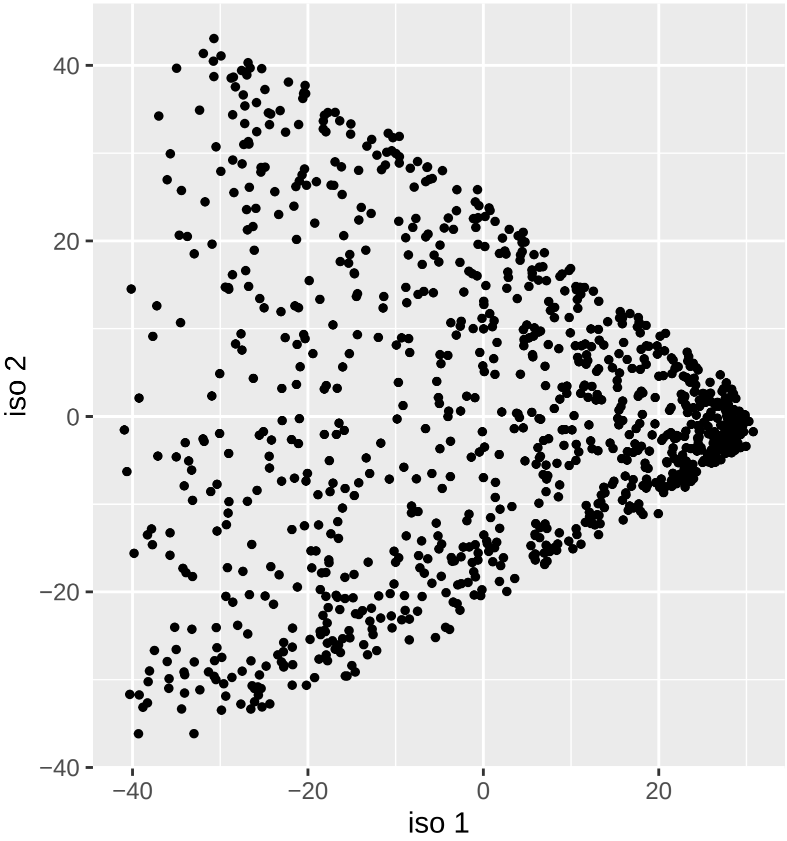
```
dfpca<-tibble(cbind(pcaout@data@meta,pcaout@data@data))

ggplot(dfpca,aes(x=PC1,y=PC2,label=Image))+
  geom_point(size=1)+coord_equal()
```



Plot for Isomap

```
dfiso<-tibble(cbind(isoout@data@meta,isoout@data@data))
ggplot(dfiso,aes(x=`iso 1`,y=`iso 2`,label=Image))+
  geom_point(size=1)+coord_equal()
```



Local continuity meta criterion

1. Compute the local continuity meta criterion (LCMC) for PCA and Isomap. Compare the LCMC across all both methods when the number of nearest neighbours used to compute the LCMC is 50. (Note this can be a bit slow)

```
lcmc_pca<-dimRed::LCMC(pcaout)
lcmc_iso<-dimRed::LCMC(isoout)
```

The LCMC is 0.2 *for PCA and* 0.22 *for Isomap.*

2. Carry out PCA using 3 PCs and recompute the LCMC.

```
pcaout3<-embed(.data = im, .method="PCA",ndim=3)
lcmc_pca3<-dimRed::LCMC(pcaout3)
```

The LCMC is 0.339 *when 3 PCs are used instead of 2 PCs.*

3. Carry out isomap using 20 nearest neighbours (and two output dimensions) and recompute the LCMC.

```
isoout20<-embed(.data = im, .method="Isomap",knn=20)

lcmc_iso20<-dimRed::LCMC(isoout20)
```

The LCMC is 0.363 *when 20 nearest neigbours are used to compute the Isomap output.*coRanking;

4. Compare the LCMC from your answers in 1-3. What do you conclude?

The LCMC is higher for Isomap than for PCA which is reasonable since Isomap is a non-linear method. If the number of output dimensions is increased for PCA the quality of the output improves. The performance of Isomap can be improved while keeping the same number of output dimensions, by choosing a different value of nearest neighbours. The main takeway messages are that a linear technique may need more dimensions to achieve the same level of accuracy as a non-linear technique and that the tuning parameter of Isomap can be quite critical.

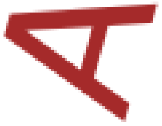
Interpreting the output dimensions

1. Using Isomap computed using 20 nearest neighbours, find the images corresponding to (i) the largest value of the first output dimension (ii) the smallest value of the second output dimension (iii) the largest value of the second output dimension. Judging, from this what may each output dimension represent?

```
dfiso20<-tibble(cbind(isoout20@data@meta,isoout@data@data))
parti<-arrange(dfiso,desc(`iso 1`))%>%head(1)%>%pull(Image)
partii<-arrange(dfiso,`iso 2`)%>%head(1)%>%pull(Image)
partiii<-arrange(dfiso,desc(`iso 2`))%>%head(1)%>%pull(Image)
```

The image corresponding to the largest value of the first output dimension

The image corresponding to the smallest value of the second output dimension



The image corresponding to the largest value of the second output dimension

The first output dimension clearly corresponds to the scale of the image (larger values indicate a smaller A). The second output may correspond to the orientation of the letter A with higher values corresponding to a more vertical orientation and larger values corresponding to a more horizontal orientation.

Bonus Question

1. Let $\mathbf{x}$ be a p-vector and $\tilde{\mathbf{X}}$ be a $p \times k$ matrix whose columns are the k nearest neighbours of $\mathbf{x}$

Prove that the value of $\mathbf{w}$ that minimises

$$||\mathbf{x} - \mathbf{w}'\tilde{\mathbf{X}}||_2^2$$

subject to $\sum w_j = 1$ is given by

$$\mathbf{w} = \frac{\sum\limits_k \mathbf{C}^{-1}_{jk}}{\sum\limits_k \sum\limits_l \mathbf{C}^{-1}_{jk}}$$

where $c_{jk} = ||\mathbf{x} - \mathbf{x}_j||_2^2$ and $\mathbf{x}_j$ is a column of $\tilde{\mathbf{X}}$.

2. Let $\mathbf{y} = (y_1, \ldots, y_n)'$ be an n-vector. Show that

$$\sum_i (y_i - \sum_j w_{ij} y_j)^2$$

is minimised subject to $\sum y_i^2 = 1$ by the eigenvector corresponding to the second smallest eigenvalue of $(\mathbf{I} - \mathbf{W})'(\mathbf{I} - \mathbf{W})$.