5 Dimensionality reduction

Aim

- Reduce the dimensionality of the spectral data for visualization and further analysis
- Eliminate noise in the spectral data

Principal Component Analysis

The classic way to deal with inter-correlations in spectral bands is to subject the data to a principal component analysis (PCA). I assume that you have done this before with spectral or other data. The PCA aims to ease the interpretation of the data while preserving the maximum amount of information. Based on the covariance matrix of the data, the original bands are transformed in principal components (PCs), which are statistically independent from each other and have a hierarchically decreasing information load. The PCs are linear combinations of the bands that use the so-called loadings of the bands as weights. By means of a PCA, we can (among others)

- reduce the dimensionality of the data to a few PCs that may be used as input variables for further modeling or analyses
- visualize the main differences between the image pixels and identify extremes
- remove noise from the image spectra

In particular for the latter, PCA is a powerful tool. Noise is a serious problem in the analysis of data with high spectral (or spatial) resolution. Noise removal is thus a crucial task in the processing of such data. Here, we will use a PCA forward-backward rotation with removal of noisy PCs, also known as minimum noise fractioning, to improve the quality of an otherwise quite noisy Hyperion data set.

The data set was acquired with the Hyperion imaging spectrometer on board the NASA EO-1 satellite. Hyperion covers the solar electromagnetic spectrum from 356 nm to 2577 nm in 242 spectral bands with a spatial resolution of 30 m on the ground (http://eo1.usgs.gov/sensors/hyperioncoverage). The sensor was launched in November 2000 and was originally planned as a one year test and demonstration mission. It was, however, continued after that period and was acquiring data until 2017. Hyperion data are in general considered noisy and require some special pre-processing. Some but not all noise can be removed by eliminating 'bad' spectral bands. Remaining artifacts can be mitigated with more advanced techniques.

Here, we want to reduce the noise in a Hyperion data set acquired in 2010 over the Fergana range in the Jalal-Abad region of Kyrgyzstan. These mountain ranges are covered with species rich walnut-wildfruit (apple, cherry, plum) forests (see, e.g. Feilhauer & Schmidtlein 2009 and Fig. 5.1). The data come as level 1T and are radiometrically and geometrically corrected. They still contain various bad bands.

Fig. 5.1: The study area near Kyzyl-Unkur in Krygyzstan.

Checking the data quality

```
library(terra)
wl <- as.vector (t (read.table ("hyperion.txt", header=F)))
im <- rast("EO1H1510312010165110KF_L1T.tif")
To save some memory we crop the image to a subset of 5 km x 5 km.
im <- crop (im, ext (335000, 340000, 4578700, 4583700))
```

Q5.1 (2 pts) Plot the image as NIR/Red/Green false color composite. Which band combination is suitable for this purpose? Briefly describe the study area in terms of forest cover, water bodies, etc.

Now we go through the data band-by-band (there are 242 bands) to visually check their quality. A for-loop further slowed down by Sys.sleep () eases this job. Some bands may actually be empty. Do not worry if the first bands give a weird result.

```
for (i in 1:242){
  plot (im[[i]], col=gray.colors (255, 0, 1, 1), main=paste (wl[i], "nm"))
  Sys.sleep (1)
}
```

You will have seen many bands that show visible noise such as speckle, stripes, and artifacts. It is often recommended to remove these bands and continue with a spectral subset. Datt et al. (2003) report 66 noisy bands and suggest to analyze the remaining 176. We follow their recommendation:

```
badbands <- c (1:7, 58:78, 121:127, 167:178, 224:242) ## Noisy bands according
                                                       ## to Datt et al. (2003)
im <- subset (im, subset=c (1:242)[-badbands]) ## remove all really bad bands
                                               ## from the image data
wl <- wl[-badbands]
```

Principal component analysis

To further mitigate the remaining noise, we apply a PCA.

Q5.2 (3 pts) The PCA can be used with and without scaling and centering the data. What is the difference between these two modes? How does this affect the influence that the individual bands get in the dimensionality reduction?

```
impix <- values (im)
pca <- prcomp(impix, center=T, scale.=T) ## Here with scaling and centering of
                                         ## the bands
```

First, we evaluate the information content of the PCs. For scaled data, an information load of < 1 SD on a PC means that a single input variable had a stronger signal.

```
sdev <- pca$sdev
plot (1:20, sdev[1:20], type="h", xlab="PC", ylab="SD")
  abline (h=1, col="#FF0000")
## plot (pca) provides a similar result, but as variances, not standard
## deviation)
```

Q5.3 (1 pt) Which PCs have an information load > 1SD?

Next, we evaluate the loadings, which quantify the importance of each band on the respective PC. Their pattern is a first indicator, whether a PC contains meaningful spectral features or mostly noise.

```
loadings <- pca$rotation # loadings indicating the band importance per PC
```

Q5.4 (3 pts) Plot wavelength vectors against the loadings of PC 1-10 and assess their distribution. Which PCs show a clear pattern, which indicate noise?

Then we take a look at the PCA-space. To ease the interpretation, we color-code the pixels according to their NDVI.

```
pcascores <- pca$x # scores indicating the pixels' position on the PCs
ndvi <- (impix[,38] – impix[,21]) / (impix[,38] + impix[,21])
cl <- colorRampPalette (c ("#ac9d93","#008000"))
ndvi <- round (255*(ndvi - min (ndvi)) / diff (range (ndvi)), 0)
plot(pcascores[,c(1,2)], pch=19, col=cl(255)[ndvi], xlab="PC 1", ylab= "PC 2")
plot(pcascores[,c(1,3)], pch=19, col=cl(255)[ndvi], xlab="PC 1", ylab= "PC 3")
plot(pcascores[,c(2,3)], pch=19, col=cl(255)[ndvi], xlab="PC 2", ylab= "PC 3")
```

Q5.5 (1 pt) How are vegetated and non-vegetated pixels distributed on the PCs?

Now we create an image of the PCs.

```
pcaim <- im
values (pcaim) <- pcascores
plotRGB (pcaim, 1, 2, 3,stretch="lin")
```

And again go through the the PCs (at least the first dozen or so) and visually check their noise level.

```
for (i in 1:176){
  plot(pcaim[[i]],col=gray.colors(255,0,1,1),main=paste("PC",i))
  Sys.sleep(1)
}
```

Minimum Noise Fractioning

To remove the noise from the image, we only keep the PCs with meaningful information and remove all others. The subset of the meaningful PCs is then reconverted to spectral bands by means of the loadings. In theory, the noise in these bands is reduced.

```
backrot <- function (sc, i) { ## sc=PCA scores, i=number of PCs to be considered
  floor (apply (t (loadings[,1:i]) * sc, 2, sum) * pca$scale + pca$center)
  }
mnfval <- t (apply (pcascores[,1:3], 1, backrot, i=3)) ## replace i=3 and 1:3 by
                    ## a different number if you want to keep more or fewer PCs
```

Q5.6 (3 pts) Plot the spectra of a few random pixels before and after the MNF-transformation. Did the shape and magnitude of the spectra change? Add a plot of a few spectra before and after to your report.

The MNF-transformed pixel values are written back to the image matrix .

```
mnfim <- pcaim
values (mnfim) <- mnfval
```

Let us do a final screening of the MNF-transformed bands to check whether they are actually showing a lower noise level.

```
for (i in 1:176){
  plot(mnfim[[i]],col=gray.colors(255,0,1,1),main=paste(wl[i], "nm"))
  Sys.sleep(1)
}
```

Q5.7 (1pt) Was the operation successful?

Isometric Feature Mapping

Isometric Feature Mapping (ISOMAP, Tenenbaum et al. 2000) is a highly non-linear dimensionality reduction. It is based on a projection of the mutual pixel-to-pixel distances in the spectral feature space to a low-dimensional ISOMAP space. To preserve and unfold non-linear structures in the data, the distance matrix is filtered and only the direct distances to k nearest neighbors are kept. Distances to remote pixels that do not belong to the nearest neighbors are newly calculates as geodesic distances following the network of nearest neighbors. By using these geodesic distances for the projection, complex structures can be unfolded and flattened (Fig. 5.2). Bachmann et al. (2005) claim that ISOMAP performs as good or even better than MNF to separate meaningful information from noise in the image data. Challenge accepted.

There are multiple implementations for ISOMAP in R, one of them being provided by Guido Kraemer that we are going to use.

```
install.packages ("dimRed")
library (dimRed)
install.packages ("RSpectra")
library (RSpectra)
install.packages ("RANN")
library (RANN)

imp <- dimRedData (impix)
```

ISOMAP requires the parameterization of the number of nearest neighbors to be considered for the geodesic distances (knn) and the number of dimensions of the solution (ndim). While the dimensionality can be chosen based on your personal requirements, knn is often tuned. This requires, however, a massive computational effort.

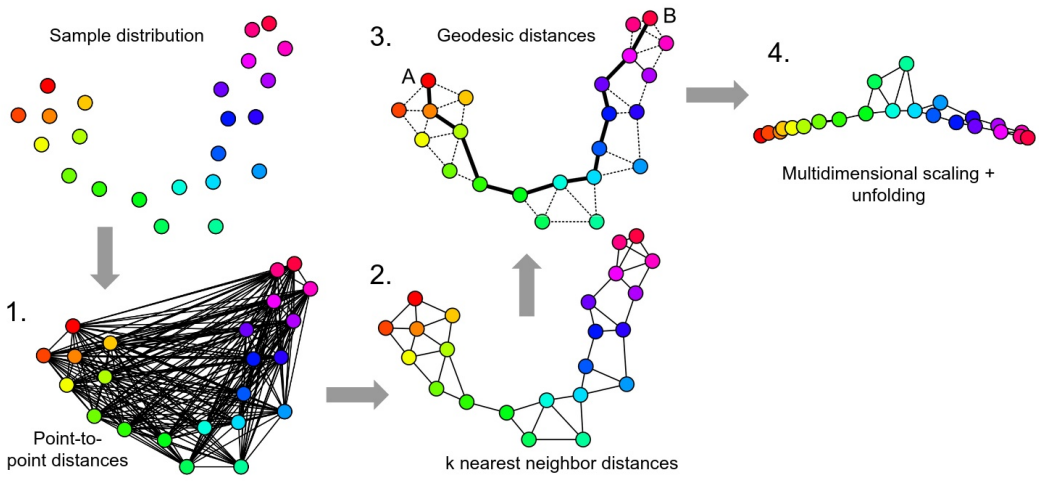

Fig. 5.2: ISOMAP geodesic distances and unfolding of a complex structure

The parameter ranges from the total number of pixels-1 to as little as 3 neighbors. In the first case, the geodesic distances are identical to the Euclidean distances and the ISOMAP solution is identical to classic multidimensional scaling. In the latter case, the nearest neighbor network is prone to being disconnected and will result in a fragmented solution. In general, relatively small knn-values will result in a non-linear solution and large values in a less pronounced unfolding. Keeping this in mind and not trying to torture our computer, we just go with knn=10 for a very non-linear setup.

```
emb <- embed (imp, "Isomap", knn = 10, ndim=3)
ismscores <- emb@data@data ## extract the scores

plot(ismscores[,c(1,2)], pch=19, col=cl(255)[ndvi], xlab="Isomap 1", ylab=
      "Isomap 2")
plot(ismscores[,c(1,3)], pch=19, col=cl(255)[ndvi], xlab="Isomap 1", ylab=
      "Isomap 3")
plot(ismscores[,c(2,3)], pch=19, col=cl(255)[ndvi], xlab="Isomap 2", ylab=
      "Isomap 3")
```

Q5.8 (2 pt) Does the ISOMAP-distribution of vegetated and non-vegetated pixels differ from the PCA-distribution?

Finally, the ISOMAP-scores are written back to the image matrix.

```
ismim <- subset (im, subset=c(1:3))
values (ismim) <- ismscores
plotRGB (ismim, 1, 2, 3, stretch="lin")
```

Q5.9 (1 pt) Is ISOMAP successfully removing the noise from the image data?

Q4.10 (3pts): Read the paper by Wang et al. and come up with one (or more) question(s) that address(es) the paper and its methodology.

Copy the answers to all ten problems and your own question(s) into a single document. You can again get a maximum of 20 points for all answers being correct and earn two extra points for for providing an individual solution. Grading will follow the scheme provided in training #2. Have fun!.

References

Feilhauer H, Schmidtlein S (2009). Mapping continuous fields of forest alpha and beta diversity. Applied Vegetation Science 12, 429–439.

Tenenbaum JB, de Silva V, Langford JC (2000). A global geometric framework for nonlinear dimensionality reduction. Science 290, 2319−2323.