Non-Negative Matrix Factorization

D2K Course Staff

2/21/2019

Non-Negative Matrix Factorization (NMF)

Just like PCA, Non-Negative Matrix Factorization is used to reduce the dimension of a dataset, visualize high-dimensional data, and find underlying patterns in the original data. It also uses linear dimension reduction (LDR) techniques in order to represent the data in a lower-dimension. In other words, we are trying to compute r basis elements such that the linear space spanned by the basis approximate the data points as closely as possible. All LDR techniques approximate the data matrix X with a low-rank matrix WH where $X \approx WH$, or in vector notation

$$x_j \approx \Sigma_{k=1}^{r} w_k h_j(k) \text{ for some weights } h_j \in \mathbb{R}^{\mathrm{r}}$$

where,

- the matrix $X \in \mathbb{R}^{\mathrm{p\times n}}$ where each column is a data point, that is, $X(:,j) = x_j$ for $1 \leq j \leq n$
- the matrix $W \in \mathbb{R}^{\mathrm{p\times r}}$ where each column is a basis element, that is, $W(:,k) = w_k$ for $1 \leq k \leq r$
- the matrix $H \in \mathbb{R}^{\mathrm{r\times n}}$ where each column of H gives the coordinates of a data point $X(:,j)$ in the basis W, that is, $H(:,j) = hj$ for $1 \leq j \leq n$

Essentially, we are representing the original data that is in p-dimensions in an r-dimensional linear subspace. The subspace is spanned by the basis elements w_k's whose coordinates are given by the vectors h_j 's. This is what allows us to express the data in a *lower dimension*.

Unlike NMF, PCA makes no assumptions about W or H. However, NMF assumes that all of the basis elements w_k's and the weights h_j's are non-negative component-wise. Thus, it is a very useful technique if you are working with non-negative data like pixel values or text data (which we will explore in this example). It allows us to automatically extract sparse features that are easy to interpret. In order to compute the linear dimension reduction, we solve the following optimization problem:

$$\underset{W\in\mathbb{R}^{p\times r}, H\in\mathbb{R}^{r\times n}}{\text{minimize}} ||X - WH||_F^2 \text{ such that } W \geq 0 \text{ and } H \geq 0$$

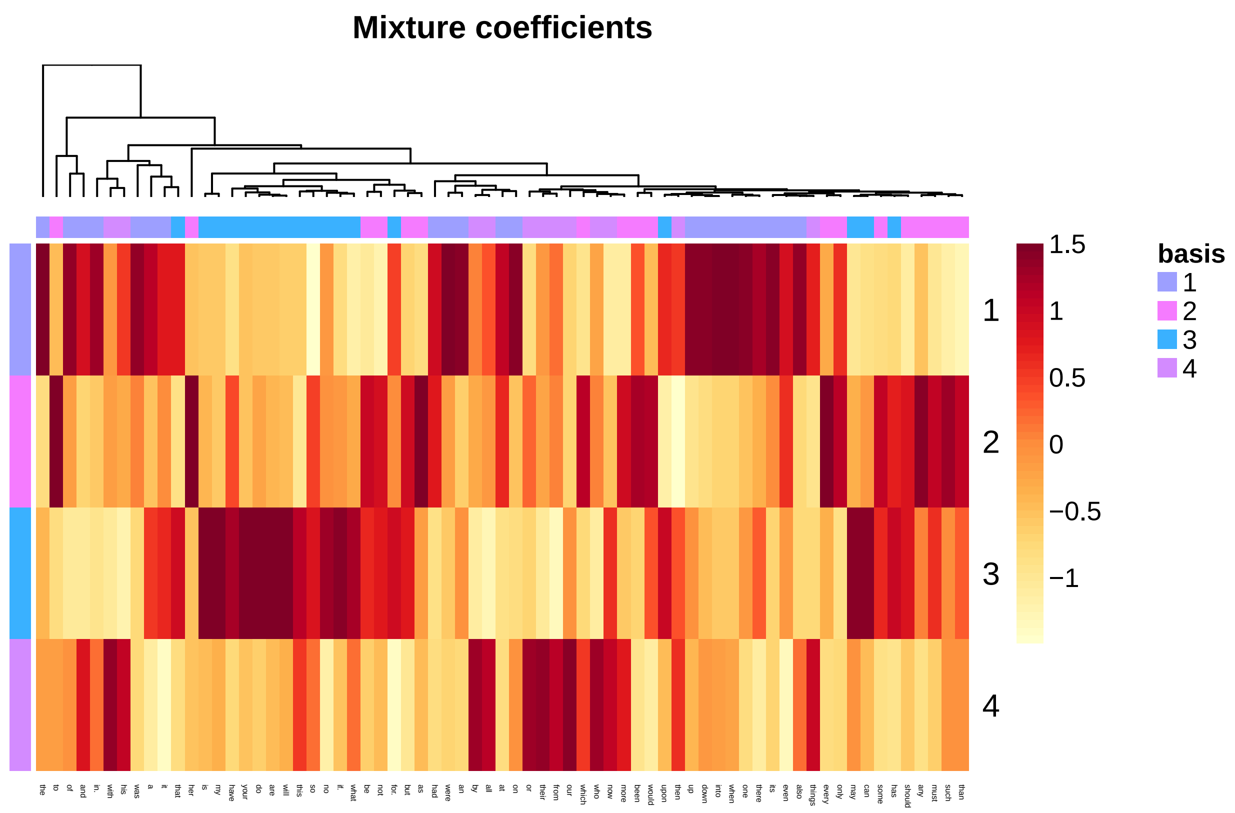
Unfortunately, unlike PCA, we cannot solve the optimization problem with singular value decomposition due to the constraints. There are, however, several solutions that we will not cover in this module. Moreover, our solution depends on the rank k of matrix W that we choose. We must carefully consider what value we choose based on what our objective is since changing k can fundamentally change factors. We strongly suggest looking at additional literature on rank selection for NMF if this is a method you are trying to apply.

We now show how to implement NMF on author text data that we are providing for this module from the `authorship.csv` file. The dataset contains word counts for several books by a handful of authors. We can see if we can see any clear patterns arise when performing NMF on this data set. Furthermore, since we know there are four different authors in the data set we wish to analyze, we choose $k = 4$ when performing NMF. We can plot a heatmap of the consensus matrix derived earlier in order to see if any clusters of features arise and any patterns become apparent using hierarchical clustering.

```r
# Load NMF library
library(NMF)
```

```
## Loading required package: pkgmaker
## Loading required package: registry

##
## Attaching package: 'pkgmaker'

## The following object is masked from 'package:base':
##
##     isFALSE

## Loading required package: rngtools

## Loading required package: cluster

## NMF - BioConductor layer [NO: missing Biobase] | Shared memory capabilities [NO: bigmemory] | Cores 

##   To enable the Bioconductor layer, try: install.extras('
## NMF
## ') [with Bioconductor repository enabled]
##   To enable shared memory capabilities, try: install.extras('
## NMF
## ')
```

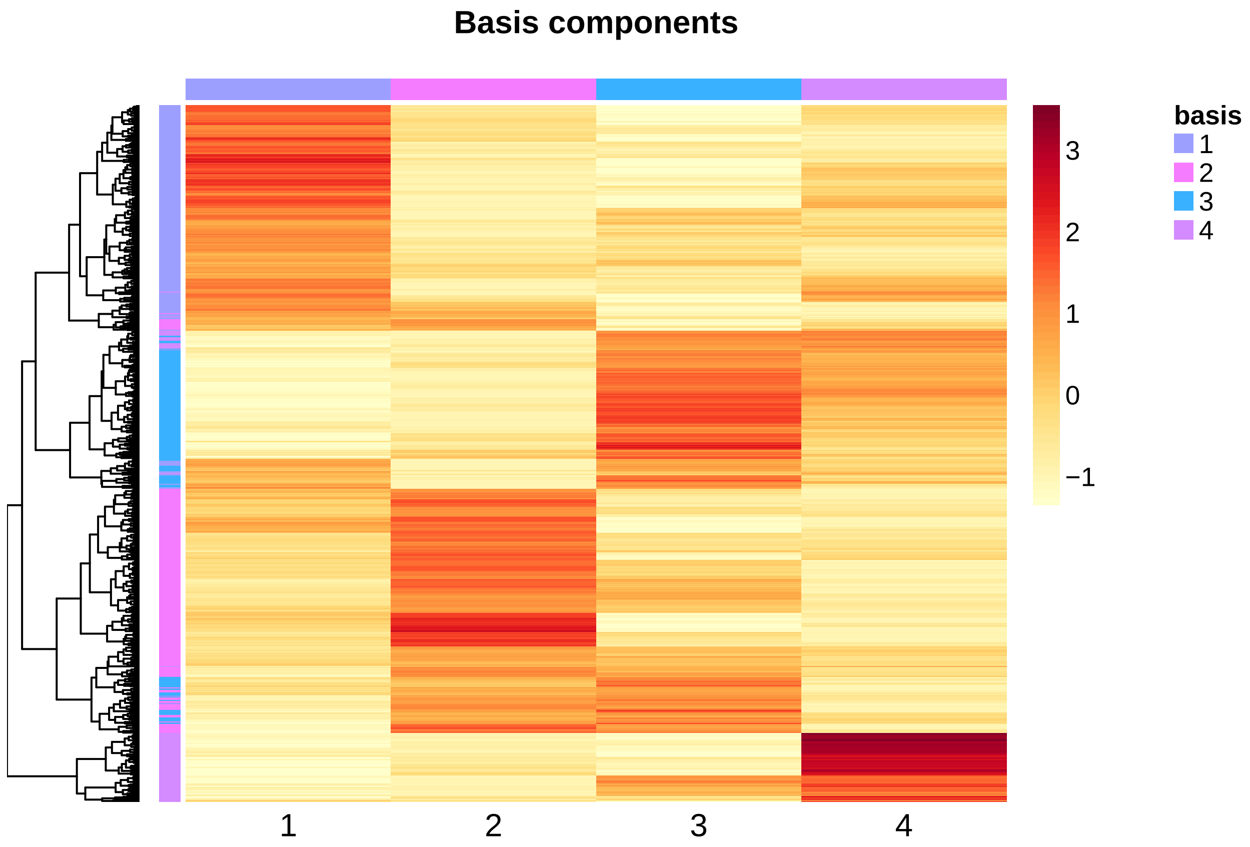
```
# Load author text data
author_df <- read.csv("data/authorship.csv", header = TRUE, stringsAsFactors = FALSE)

# Separate data into features and labels
author_x <- as.matrix(author_df[,-c(70:71)])
author_labels <- as.factor(author_df[,71])

# Create matrix for analysis
X <- author_x
rownames(X) <- author_labels

# NMF
K <- 4
nmffit <- nmf(X,rank=K)
W <- basis(nmffit)
H <- coef(nmffit)

# Visualize coefficient matrix
coefmap(nmffit, scale="col", legend=TRUE)
```



We can see how each feature contributes to the low-dimensional representation of the data with this heatmap. Darker colored cells in the heat map correspond to greater values in coefficient matrix for that particular basis vector. This allows us to see which features should possibly be considered together in future analyses.

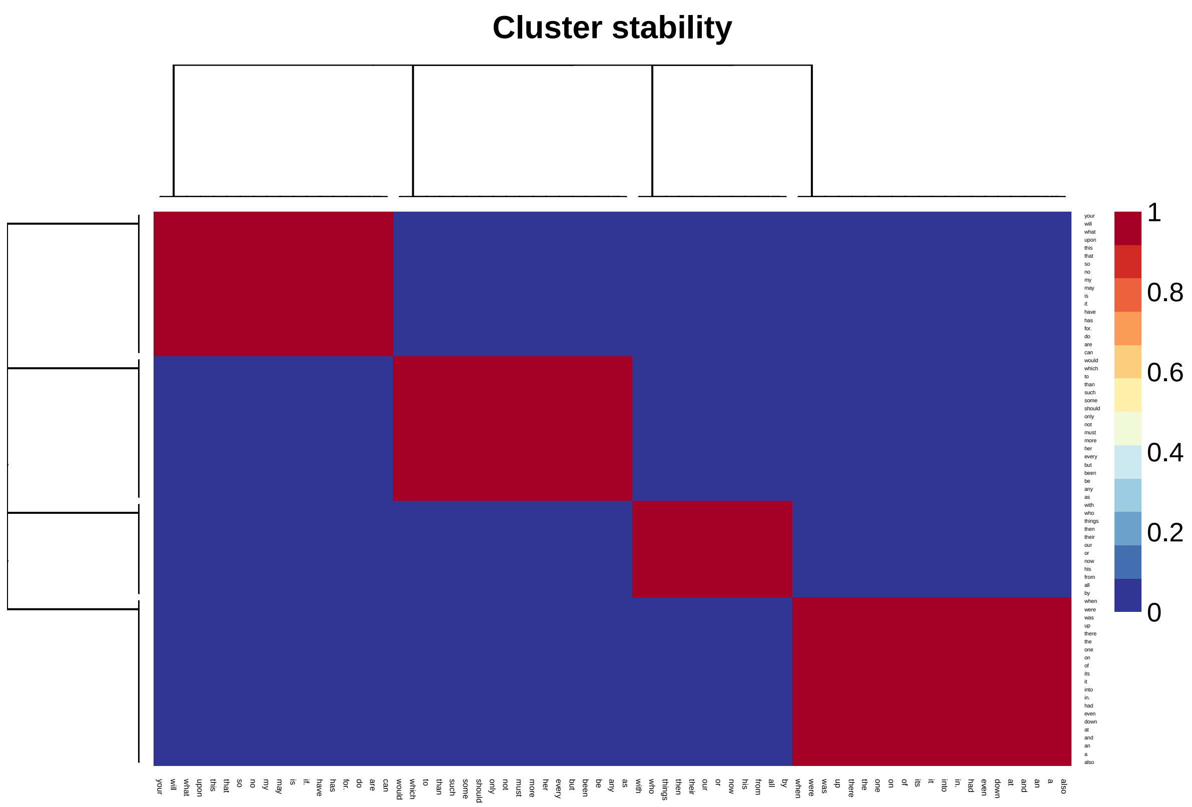
We can also visualize the basis matrix on a heatmap with hierarchical clustering as well.

```
# Visualize basis matrix
basismap(nmffit, scale="col", legend=TRUE)
```



And finally we can visualize the consensus matrix over multiple independent NMF runs with a heatmap to understand the stability of the clusters we previously observed.

```
consensusmap(nmffit, main='Cluster stability')
```



By doing this, we can see 4 distinct clusters of words from our original data set.

Advantages

- Works with non-negative data
- Pattern recognition
- Works well for text-data and image data

Disadvantages

- Strong assumptions about data
- Can only extract linear patterns

References

Gaujoux, Renaud. "Generating Heatmaps for Nonnegative Matrix Factorization." CRAN, 6 Feb. 2018, cran.r-project.org/web/packages/NMF/vignettes/heatmaps.pdf.

Gillis, Nicolas. "The Why and How of Nonnegative Matrix Factorization." ArXiv, Department of Mathematics and Operational Research Facult´e Polytechnique, Universit´e De Mons, 7 Mar. 2014, arxiv.org/pdf/1401.5226.pdf.