Multiway Lecture - Casein Exercise (for Python)

April 23, 2020

Computational Data Analysis

Course 02582

Andreas Baum, andba@dtu.dk

NOTE: Please do not re-distribute the data used in this exercise!

Required R packages: *dill*, *numpy*, *scikit-learn*, *tensorly*, *matplotlib*

1 Background

Fourier Transform Infrared spectroscopy (FTIR) is used in chemistry as a rapid analytical method to obtain a snapshot representing the entire chemical complexity of a given sample.

The present data describes FTIR spectra of milk samples which were measured time-resolved during an ongoing enzymatic reaction. A protease - an enzyme which is capable of degrading proteins - was added to each sample containing varying casein concentrations (a milk protein). As soon as the protease was injected FTIR spectra were acquired consecutively. Due to the enzyme activity the spectra started to change. The resulting spectral evolution represents the kinetic behavior of the enzymatic reaction monitored. Such spectral evolution profiles (eps) were acquired for each sample given different initial casein concentrations.

Each evolution profile (ep) is represented as a $N \times M$ data matrix with N temporal time points (= number of spectra) and M spectral wavenumbers. Some data pre-processing has been performed for all eps, i.e. the first spectrum of each ep was subtracted from each respective series. Hence, the spectral changes appear in reference to the initial spectrum at time point 0.

The goal of this exercise is to establish a suitable PARAFAC model which is capable of quantifying the initial casein concentration. No Cross-Validation should be performed, instead the entire data shall be considered in an unsupervised learning regime.

Data reference: Baum et al., *Journal of dairy science* 99, no. 8 (**2016**): 6071-6079.

2 Questions

2.1 Install all required Python3 packages and load the data! (You may require upgraded packages)

Load the pickled data (`casein.pkl`) using dill. Each list entry in `eps_list` represents a spectral evolution profile (ep) for a given casein concentration.

[ ]:
```
import dill
import pandas as pd
import numpy as np
from matplotlib import pyplot as plt

dill.load_session('casein.pkl')
# Form tensor x by stacking list of matrices (eps)
x = np.array(eps_list)
# casein_conc contains the known casein concentrations for the 36 samples
print(len(casein_conc))
# scale_temporal and scale_spectra contains the axis labels for the temporal and␣
 ↪spectral mode
print(len(scale_temporal))
print(len(scale_spectra))
```

2.2 What are the dimensions of the tensor $\mathcal{X}$? What are the three modes of the tensor?

[ ]:
```

```

2.3 Visualize some of the eps! You may use the function provided below.

Each ep can be represented as a tensor slab (matrix). Hence, eps can be visualized using 3d surface plots. Let's create a function for convenient surface plotting of eps.

[ ]:
```
def surf_plot_comparison(eps_sel, casein_sel):
    # A function for convenient comparison of several eps
    from mpl_toolkits.mplot3d import Axes3D
    from matplotlib import cm
    fig = plt.figure(figsize=(15,10), dpi=600)
    for pos in range(len(eps_sel)):
        ax = fig.add_subplot(2, 3, pos+1, projection='3d')
        ax.plot_surface(*np.meshgrid(scale_spectra, scale_temporal),
                        eps_sel[pos], cmap=cm.jet)
        ax.view_init(25,235)
        ax.set_zlim(eps_sel.min(), eps_sel.max())
        ax.set_xlabel('$\\lambda$ in $cm^{-1}$')
```

```
        ax.set_ylabel('time in $min$')
        ax.set_zlabel('$Abs$')
        ax.set_title('$c_{casein} = %.2f \; g/L$' % casein_sel[pos])
```

[ ]:

2.4 **For the following assume the notation of the decomposition as $\mathcal{X} = A(B \odot C)^T + \mathcal{E}$, where $\mathcal{X}$, A, B, C and $\mathcal{E}$ represent the three-way tensor, sample mode loadings (PARAFAC scores), temporal mode loadings, spectral mode loadings and residual tensor, respectively and $\odot$ represents the Khatri-Rao product. Decompose the three-way tensor using PARAFAC. How many components do you require? Why?**

[ ]:
```
from tensorly.decomposition import parafac
```

2.5 **Compute the outer vector product of the first component's temporal and spectral loading ($b_1 c_1^T$). Surf plot the result. How does the plotted result relate to the plotted eps from the first figure?**

You may use the function `surf_plot` given below.

[ ]:
```
def surf_plot(ep_sel):
    # A function for convenient surf plotting
    from mpl_toolkits.mplot3d import Axes3D
    from matplotlib import cm
    fig = plt.figure(figsize=(10,8), dpi=600)
    ax = fig.add_subplot(1, 1, 1, projection='3d')
    ax.plot_surface(*np.meshgrid(scale_spectra, scale_temporal),
                    ep_sel, cmap=cm.jet)
    ax.view_init(25,235)
    ax.set_xlabel('[$cm^{-1}$]')
    ax.set_ylabel('[$min$]')
    ax.set_zlabel('[$Abs$]')
```

[ ]:

2.6 **Reconstruct the tensor using A, B and C, such that $\hat{\mathcal{X}} = A(B \odot C)^T$. You may simply use the function below. Thereafter, compute the residuals $\mathcal{E} = \mathcal{X} - \hat{\mathcal{X}}$. Look at some of the residuals using similar surface plots as in question 3.**

[ ]:

2.7 Unfold the tensor appropriately and perform PCA using one component!

[ ]:
```
from sklearn.decomposition import PCA
```

2.8 Scatter plot the PCA and PARAFAC scores against the casein concentration. What do you observe? Are these unsupervised models appropriate to quantify the casein concentration in milk? How many parameters were fitted using PCA? How many using PARAFAC? What is the advantage of PARAFAC in this application?

[ ]:
```

```