Tensor Decomposition Project

Gaute Johannessen

June 2025

Part 1

In this part, we analyze a third-order tensor using a CP-model. Our dataset $\mathcal{X} \in \mathbb{R}^{28\times 251\times 21}$ represents measurements of mixtures measured using fluorescence spectroscopy. The three modes of the tensor are as follows: 28 mixtures in the first mode, emission wavelengths in the second, and excitation wavelengths in the third. We will use CP-models to reveal the number of chemicals in the mixtures, and find properties of the chemicals by plotting the discovered factors in the model. In addition to this, we will discuss the uniqueness of the CP-model by comparing multiple models achieving similar losses.

Background and Method

We fit the CP-model to the data using the `cp_wopt` function from the Tensor Toolbox [3]. The reason we have selected this function is that the data having missing values. Indeed, the function allows us to pass a weight tensor $\mathcal{W}$ as an argument, where we define $\mathcal{W}$ as in (1).

$$\mathcal{W}_{i,j,k} = \begin{cases} 1 \text{ if } \mathcal{X}_{i,j,k} \text{ is in the dataset,} \\ 0 \text{ if } \mathcal{X}_{i,j,k} \text{ is missing.} \end{cases} \tag{1}$$

$$\min_{\hat{\mathcal{X}}} \frac{1}{2}||\mathcal{W} * (\mathcal{X} - \hat{\mathcal{X}})||^2 \text{ where } \hat{\mathcal{X}} = \sum_{r=1}^{R} \lambda_r \mathbf{a}_r \circ \mathbf{b}_r \circ \mathbf{c}_r \tag{2}$$

We use this weight matrix to define the expression we want to optimize, described in (2). This is the same description of the CP-model as used by Kolda and Bader [5], with the added weight tensor for handling the missing data, described by Acar et al. [1]. In (2), the norm used is the Frobenius norm for tensors, $*$ denotes the element-wise Hadamard product, and $\circ$ denotes an outer vector product. This weight matrix formulation allows us to fit the model based on all our data, without having to replace the missing data with any synthetic replacements.

Having set up the CP-model (2), we now need a way to fit the optimal $\hat{\mathcal{X}}$, i.e. finding the rank R, weights $\boldsymbol{\lambda}$, and matrices A, B and C. Note that $\mathbf{a}_r$ is the rth column of A, and similar for $\mathbf{b}_r$ in B and $\mathbf{c}_r$ in C. The `cp_wopt` algorithm precomputes $\mathcal{Y} = \mathcal{W} * \mathcal{X}$ for efficiency, and computes the gradients of the function with respect to the matrices A, B and C respectively. The gradient of f with respect to A is included in (3) as an example [1]. Here, f refers to the function to be minimized as described in 2, $\mathcal{Z}$ is defined as $\mathcal{Z} = \mathcal{W} * \hat{\mathcal{X}}$, $\odot$ denotes the Khatri-Rao matrix product (see Kolda and Bader [5], p. 462), and $Z_{(n)}$ denotes the n-mode matricization of the tensor $\mathcal{Z}$.

$$\frac{\partial f}{\partial A} = (Z_{(1)} - Y_{(1)})(C \odot B) \tag{3}$$

For the optimization step, we use the L-BFGS-B algorithm, which is a version of the Limited-memory Broyden–Fletcher–Goldfarb–Shanno algorithm (L-BFGS) made for solving large nonlinear optimization with simple bounds on the variables [7]. This is the recommended optimization algorithm for the `cp_wopt` function according to the Tensor Tooblox documentation. We also impose a non-negativity constraint in all modes, as we expect the patterns in the underlying data to be non-negative.

As the loss landscape of the model is expected to be non-convex, the initial values of the model used in fitting could affect the final result, e.g. if we start close to a local minima, we could converge to that

instead of the global minima. To ensure that we find a good model for the problem, we initialize the model multiple times using random initialization before we fit each of them. For further analysis, we will always keep the best model, meaning the model with the lowest loss after its final iteration. We will also use multiple initializations to illustrate the uniqueness of CP models. Indeed, we will keep all models performing (almost) as well as the optimal model found (for part 1 we will use $f \leq 1.02 \cdot f_{opt}$, i.e. all models with loss less or equal to 2% more than the best model) and use the `score` function from the Tensor Toolbox [3] to measure compare the factors from the different runs. The `score` function computes the Factor Match Score (FMS) between two proposed models, where a score of 1 means that the models are identical [2].

The final important concept to discuss before the result is how we find the correct model rank, i.e. what value should R have in (2). If the data we want to model has a natural number of factors, using this as the value of R is likely to give the best and most interpretable model. In our case, the data is measurements of mixtures containing a number of chemicals, hence having R equal the number of distinctive chemicals would make sense. However, we do not know in advance how many chemicals there are in the mixtures. Hence, we need a way to determine the correct underlying rank.

As discussed by Bro and Kiers [4], a common way to do this is looking at loss values and relative fit (RelFit) scores for different values of R, and choosing the point where the improvement of these two scores slows down. The relative fit is a measure of percentage of explained variation (see Bro and Kiers [4], page 277 for definition). An important observation to note is that both the loss and RelFit are likely to improve past the optimal R; the clue is to find where the improvement is radically slower. Bro and Kiers [4] points out that since it is sometimes hard to find this point, as well as expensive to train multiple models for each R to get trustworthy results, this method could be bad. That said, in our case the models are relatively quick to fit, and we can confirm our pick for R using factor plots, as will be discussed in the result section. Hence, it is not necessary to use a more complex method for finding R such as the proposed CORCONDIA method [4].

Results and Discussion

As discussed in the previous section, we first find the optimal rank R for the model. We did this by fitting 20 models with different random initializations for $R \in \{1, 2, 3, 4, 5\}$, and computing the loss and RelFit for the best model for each R. The results are shown in Table 1. Note that the loss is relative to the loss of the best model with rank $R = 1$.

R	Loss	RelFit
1	1	77.589
2	0.0975	96.497
3	0.0065	98.891
4	0.0045	98.936
5	0.0028	98.967

Table 1: Loss relative to loss with rank $R = 1$ and RelFit for $R \in \{1, 2, 3, 4, 5\}$ for best model of 20 runs for each R.

From Table 1, we see that the loss is decreasing by a factor greater than 10 for each R until $R = 3$, after which it decreases by a factor less than 2. Furthermore, the relative fit increases substantially from $R = 1$ to $R = 2$ and $R = 2$ to $R = 3$, after which the growth is significantly slower. Hence $R = 3$ seems like a good choice of rank, and we use this for further analysis. Figure 1 confirms this choice, as we see that the factors for $R = 3$ seems to represents three different chemicals, while for $R = 4$, the red and green factor seems to both represent the same chemicals, as they are more or less identical in the emission and excitation modes.

Next, we ran the model with $R = 3$ as rank 20 times, in this case saving all the obtained models. Comparing the models, we saw that all models reached the exact same loss. Furthermore, they all achieved an FMS of 1 when compared to the first model using the `score` function. That is, all models ended with the exact same fit regardless of initialization, which demonstrates that the uniqueness of the factors revealed by the CP decomposition. As a sanity check, we did the same analysis for $R = 4$, where we only had 17 out of the 20 models within 2% of the loss, and most models achieved an FMS of less than 0.95 when compared to

the best model. This finding confirms that the exact matches discovered for rank $R = 3$ is not due to some error in our method of comparison.

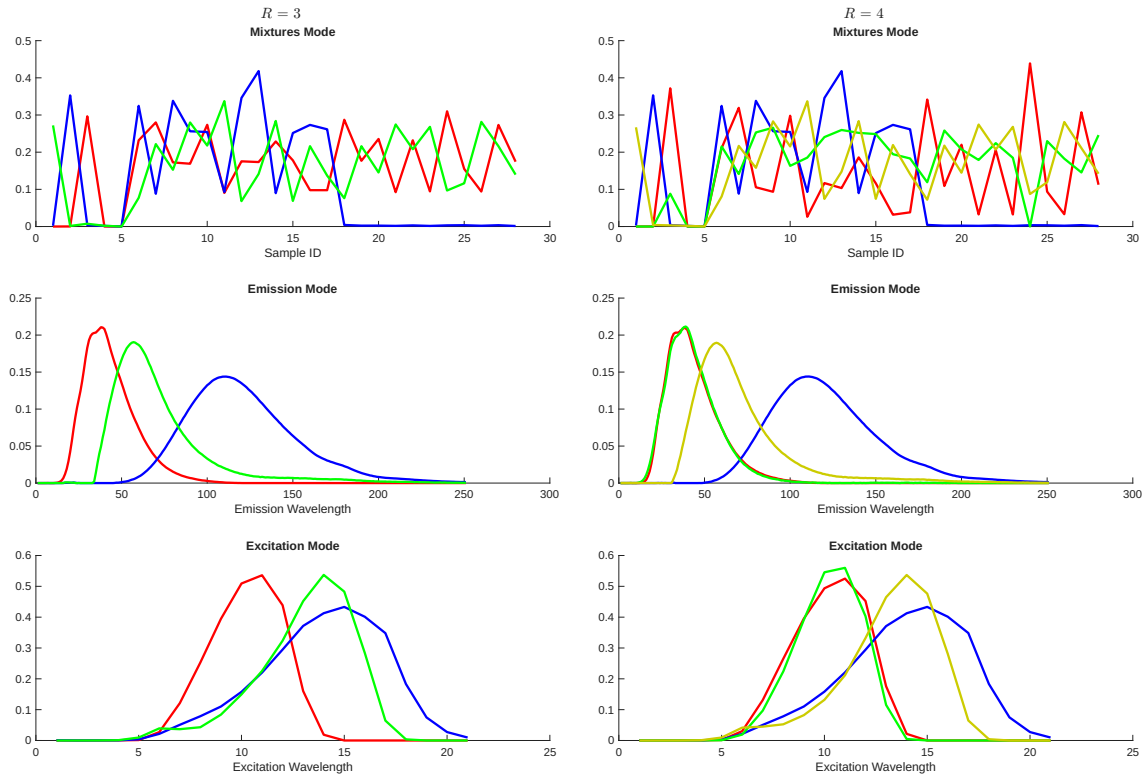

Figure 1: Plots of normalized factors for each mode from the best CP-model over 20 runs for $R = 3$ (left) and $R = 4$.

Lastly, we plotted the factors in each mode for the best model for $R = 3$ and $R = 4$, as seen in Figure 1. We plotted $R = 4$ only to confirm our choice of R, as discussed above. For the $R = 3$ plots, we see that the model seems to have discovered three different chemicals. The top left plot, the mixtures mode, represents the amount of each chemical in each of the 28 mixtures, while the plots below represent emission and excitation wavelengths for each chemical. It is expected that the top plot is less smooth than the two below, as there is no reason why the amount of each chemical in each mix should be a smooth function. For the wavelengths however, it is expected that the plot for each chemical is quite smooth.

Part 2

In this subsection, we jointly analyze measurements from multiple sources in order to reveal underlying patterns across them. The datasets consist of fluorescence spectroscopy EEMs $\mathcal{X} \in \mathbb{R}^{28\times 251\times 21}$, identical to part one, as well as nuclear magnetic resonance (NMR) spectroscopy $\mathcal{Y} \in \mathbb{R}^{28\times 13324\times 8}$ and liquid chromatography-mass spectrometry (LC-MS) measurements $Z \in \mathbb{R}^{28\times 168}$, all from the same 28 mixtures.

We will jointly analyze the three datasets using the AO-ADMM Data Fusion Framework [6], demonstrate the uniqueness property of the model by comparing different runs, and finally plot the discovered factors.

Background and Method

As in part 1, our first task is to handle the missing values in $\mathcal{X}$. We note that neither $\mathcal{Y}$ nor Z contain any missing values. While the `cp_wopt` function we used in part 1 has a simple way of handling missing values internally by weighting the loss function, this is not the case for the method used in part 2. Hence,

we need to handle these missing values before fitting the models. There are various methods to do this, but we chose to use the best model from part 1 to replace the missing values. This seems like a good solution, as the model gives replacement values based on the entire rest of the dataset, instead of a weighted average of some selected values, as is the case for some alternatives. Instead of using $\mathcal{X}$ directly, we fitted the coupled model using $\mathcal{X}^*$ defined as in (4) instead, where $*$ denotes the element-wise Hadamard product, $\hat{\mathcal{X}}$ is the best $R = 3$ model from part 1, $\mathcal{W}$ is defined as in (1), and $\mathcal{W}'$ is defined as $\mathcal{W}'_{i,j,k} = 1 - \mathcal{W}_{i,j,k}$.

$$\mathcal{X}^* = \mathcal{W} * \mathcal{X} + \mathcal{W}' * \hat{\mathcal{X}} \tag{4}$$

We analyze the data jointly using the AO-ADMM Data Fusion Framework[6]. This framework utilizes Alternating Optimization (AO) and the Alternating Direction Method of Multipliers (ADMM). For more information about these algorithms, see Schenker, Cohen, and Acar [6]. We use the Frobenius tensor norm as the loss function on all three matrices, and L-BFGS-B as the optimization algorithm.

Based on the information given about the underlying components, we use the coupling set-up described in case 3b in Schenker, Cohen, and Acar [6]. As the total number of components is 6 (3 across all tensors, 1 across $\mathcal{Y}$ and Z, 1 only in $\mathcal{Y}$ and one only in Z), we let the matrix $\Delta_1 \in \mathbb{R}^{28\times 6}$. The factor 28 comes from all 28 mixtures being coupled across all three tensors/matrices. By letting $C_{1,1}$ denote the factor matrix of the first mode in $\mathcal{X}$, $C_{2,1}$ the first mode in $\mathcal{Y}$, and $C_{3,1}$ in Z, we have $C_{i,1} = \Delta_1 H_{i,1}$ for each $i = 1, 2, 3$, where $H_{i,1}$ are defined for each i as in (5).

$$H_{1,1} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \quad H_{2,1} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}, \quad H_{3,1} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{5}$$

Note that from the matrices in (5), we see that the $\mathcal{X}$ will be modeled by the first 3 factors, $\mathcal{Y}$ by components 1 through 5, and Z by components 1 to 4 and 6, as described in the task.

We also note that the matrix $C_{1,1} = \Delta_1 H_{1,1}$ will be of size 28×3, as expected. Similarly, both $C_{2,1}$ and $C_{3,1}$ will be of size 28×5.

As in part 1, we will perform multiple fitting runs with randomly drawn initial conditions in order to obtain the best model possible. We also compare the models that have (almost) equal loss to the loss of the best model, in order to demonstrate uniqueness of the factors.

Results and Discussion

After fitting the models 30 times, we consider the models with a loss equal to at most 1.0005 times the loss of the best model. There are two models (apart from the best model) with low enough losses to be considered, and their match scores compared with the best model for each of the three datasets are shown in Table 2 below.

$\mathcal{X}$	$\mathcal{Y}$	Z
0.9996	0.9997	0.9959
1.0000	1.0000	0.9731

Table 2: FMS scores for second and third best models compared to best model.

We see that the factors for every part of the dataset have FMS scores close to 1, demonstrating that models with loss close to the best model also have their factors close to the best model. This demonstrates uniqueness: we do not obtain models of similar performance to the best one with vastly different sets of factors.

All factors from the CMTF-analysis are plotted in Figure 2 (EEM and NMR) and Figure 3 (LC-MS). We notice that the factors of the EEM part in Figure 2 closely resemble those in Figure 1. This is to be expected, as they model the same data. Furthermore, we see that the first three factors are identical in the

first mode over all three plots, as well as the fourth factor (yellow line) being identical in the first mode for NMR and LC-MS plots. This confirms that our couplings works as expected.

As in part 1, we can identify the underlying properties of each chemical contained in the mixtures by studying the plots. Indeed, the chemical shifts, gradient levels, and features of each of the chemicals are shown in Figures 2 and 3.

Note that the relative sizes of the factors are slightly different between Figures 1 and 2, since the factors in Figure 2 are not normalized. When comparing, we are only interested in the shapes of the graphs, not the absolute size.

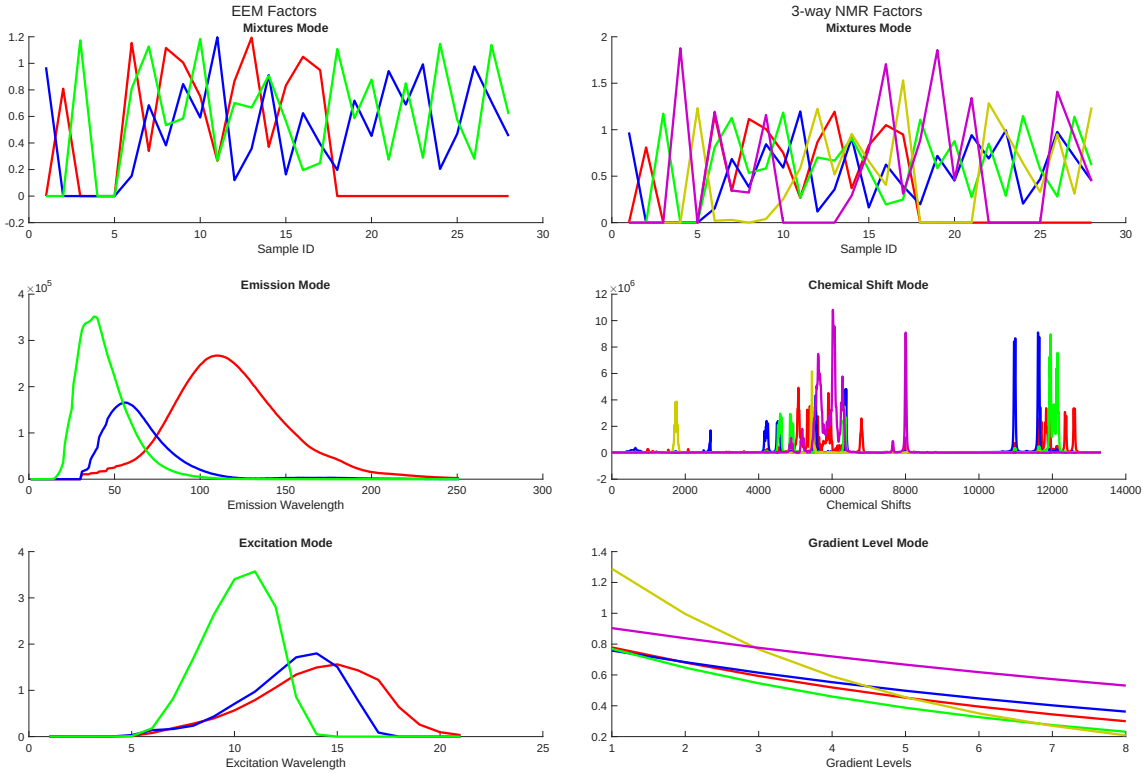

Figure 2: Discovered factors for EEM and NMR data

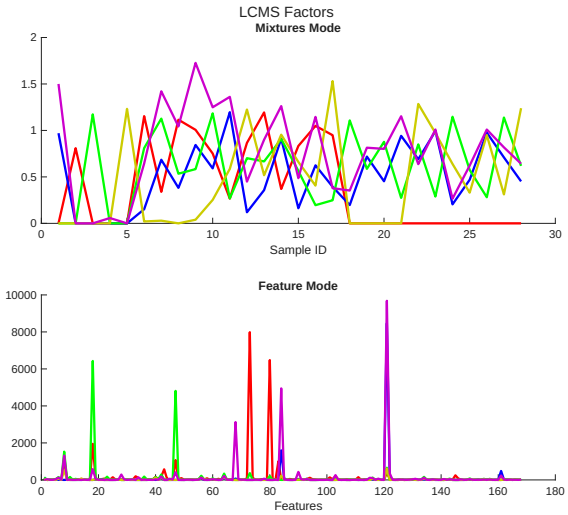

Figure 3: Discovered factors for LC-MS data.

Conclusions

The two parts of this project show how we can utilize tensor decompositions and CMTF to analyze different datasets and discover the underlying patterns. In part 1, we were able to not only discover the number of relevant chemicals in the mixtures, but also gather information about their properties, which in turn would allow us to identify the chemicals if they were unknown. Part 2 demonstrates how coupled tensor matrix factorizations can be used to discover common factors across multiple datasets coupled in a single mode. This lets us explore more complicated underlying patterns across data gathered through different experiments. The uniqueness property of the models ensures that the patterns found are indeed the "correct" patterns, and not a random set of factors that optimizes the mathematical problem.

Appendix: Code

All results discussed can be replicated in MatLab using the code in this GitHub repository. For further details about packages, versions, and how to run the code, please refer to the ReadMe file in the GitHub repository.

References

[1] Evrim Acar et al. "Scalable tensor factorizations for incomplete data". In: *Chemometrics and Intelligent Laboratory Systems* 106.1 (2011). Multiway and Multiset Data Analysis, pp. 41–56. ISSN: 0169-7439. DOI: https://doi.org/10.1016/j.chemolab.2010.08.004. URL: https://www.sciencedirect.com/science/article/pii/S0169743910001437.

[2] Tülay Adali et al. "Reproducibility in Matrix and Tensor Decompositions: Focus on Model Match, Interpretability, and Uniqueness". en. In: *IEEE Signal Process Mag* 39.4 (June 2022), pp. 8–24.

[3] Brett W. Bader, Tamara G. Kolda, et al. *Tensor Toolbox for MATLAB*. Version Version 3.6. URL: www.tensortoolbox.org.

[4] Rasmus Bro and Henk A. L. Kiers. "A new efficient method for determining the number of components in PARAFAC models". In: *Journal of Chemometrics* 17.5 (2003), pp. 274–286. DOI: https://doi.org/10.1002/cem.801. eprint: https://analyticalsciencejournals.onlinelibrary.wiley.com/doi/pdf/10.1002/cem.801. URL: https://analyticalsciencejournals.onlinelibrary.wiley.com/doi/abs/10.1002/cem.801.

[5] Tamara G. Kolda and Brett W. Bader. "Tensor Decompositions and Applications". In: *SIAM Review* 51.3 (Sept. 2009), pp. 455–500. DOI: 10.1137/07070111X.

[6] Carla Schenker, Jeremy E. Cohen, and Evrim Acar. "A Flexible Optimization Framework for Regularized Matrix-Tensor Factorizations with Linear Couplings". In: *CoRR* abs/2007.09605 (2020). arXiv: 2007.09605. URL: https://arxiv.org/abs/2007.09605.

[7] Ciyou Zhu et al. "Algorithm 778: L-BFGS-B: Fortran subroutines for large-scale bound-constrained optimization". In: *ACM Trans. Math. Softw.* 23.4 (Dec. 1997), pp. 550–560. ISSN: 0098-3500. DOI: 10.1145/279232.279236. URL: https://doi.org/10.1145/279232.279236.