Министерство образования и науки Российской Федерации
Московский физико-технический институт (государственный университет)
Физтех-школа прикладной математики и информатики
Кафедра «Интеллектуальные системы»

Алсаханова Надежда Юрьевна

Выбор тензорных представлений для прогнозирования по мультимодальным измерениям.

03.04.01 — Прикладные математика и физика

Выпускная квалификационная работа магистра

Научный руководитель:
д. ф.-м. н. Стрижов Вадим Викторович

Москва
2023

Contents

Аннотация

Решается задача выбора модели прогнозирования квазипериодических временных рядов. Классические методы оценки эффективности прогнозирования не учитывают предысторию и априорное знание о периодичности временных рядов. Для учета этой информации применяется модель фазовой траектории исследуемого временного ряда. Для перехода в фазовое пространство временной ряд векторизуется с помощью метода задержек. Для понижения сложности модели в фазовом пространстве выбирается подпространство. Проекция фазовой траектории на сферу в полученном подпространстве аппроксимируется линейной комбинацией сферических гармоник. Полученная модель оценивает вероятность принадлежности точек фазовой траектории исследуемому временному ряду. Дополнительно свойства модели фазовой траектории были исследованы в задаче классификации. По полученным оценкам вероятности предлагается метод выбора наилучшей модели прогнозирова Вычислительный эксперимент проведен на измерениях акселерометра мобильного устройства с тремя классами движений человека.

Ключевые слова: выбор моделей, временные ряды, прогнозирование, аппроксимация, классификация, фазовое пространство, сферические гармоники.

Abstract

Brain-computer interfaces (BCI) are a rapidly developing research area that can help people to restore their movement ability. Modern BCI data are multiview, as they have many measured signals. These brain signals are redundant and highly correlated in both temporal and spatial modes. It leads to unstable models and inaccurate predictions. Therefore, dimensionality reduction methods are used.

Many dimensionality reduction methods cannot take into account correlations in temporal and spatial modes. To reveal these correlations, tensorization is used before the reduction of dimensionality. One of the most common method of tensorization for time series is hankelization. It can incorporate intrinsic temporal and spatial correlations. However, this method has not been used in the forecasting problem in the area of BCI.

In this work, we found optimal representations of feature and target tensors in latent space by combining hankelization and dimensionality reduction methods. We tested hankelization along the temporal, along the spatial, and along both dimensions. In addition, we used tree of the most common and modern dimensionality reduction models to check whether hankelization works for all models. These models are multilinear principal component analyses (MPCA), high-order partial least squares (HOPLS) and Hybrid Autoencoder.

The results on three BCI datasets showed that hankelization along the space dimension improves prediction quality the most and simplifies latent tensor for many cases. Additionally, hankelization along both dimensions works the best way for some cases. These results show that hankelization can improve forecasting quality in the BCI area.

Key words: hankelization, signal decoding, ECoG, partial least squares, hybrid autoencoder, tensor regression.

1 Introduction

An initial data in the brain-computer interface's (BCI) task typically has attributes of high dimensionality, strong input correlation, and low signal-to-noise ratio. Dimensionality reduction methods are used to reduce redundancy of this data and so increase the speed of algorithms and their quality. It has previously been observed that tensorization, the generation of higher-order structured tensors from the lower-order data formats, helps to find low-rank approximation with a high level of compression and to reveal hidden correlations [3], [5], [11], [25]. There are many methods of tensorization that can be used in BCI. One of them is hankelization, construction of low-rank Hankel matrix (tensor) from vector (matrix). The Hankel matrix is a structured matrix with constant entries in each anti-diagonal. There are also methods for constructing Hankel tensors from matrices [22]. Hankelization is a method that helps to reveal hidden correlation in temporal and spatial dimensions. Several attempts have been made to implement hankelization before the prediction of time series in different tasks [20], [9], [18], [17]. However, no previous study has investigated the influence of hankelization for the BCI area.

The thesis goal is to find optimal representations of feature and target tensors in latent space by combining hankelization and dimensionality reduction methods. These tensors representations should be optimal in terms of forecasting quality of the target variables and complexity of methods.

Objectives:

- determine whether hankelization along temporal mode improves quality of forecasting;
- determine whether hankelization along spatial mode improves quality of forecasting;
- determine whether hankelization along both modes improve quality of forecasting;

The BCI datasets usually contain simple target variable as classes of some movements. However, there are several datasets in which the target variable

is a set of time series. These time series can be correlated and redundant as well as independent variable. One of the most popular datasets for forecasting task is food-tracking datasets by project Neurotycho [2], [19]. These datasets consist of epidural and subdural electrocorticography (ECoG) signals from monkeys. In addition, more and more human datasets for BCI forecasting are becoming available. For example, a human electroencephalography (EEG) dataset [8] has 6-channel electromyography signals from arms.

In the datasets for BCI, the brain signals have been taken from a large number (> 30) of closely spaced electrodes. Such excessive dimensionality of the feature description leads to instability of models. Consequently, a great deal of previous research into BCI has focused on dimensionality reduction techniques [12]. One of the most commonly used dimensionality reduction methods is principal component analyses (PCA) and its modifications (Multilinear PCA [13]). PCA and its modifications capture the greatest variance in the input data. However, it also includes the variance in noise signals.

All the above mentioned datasets have not only correlated and noisy brain signals but also correlated target variables. Thus, dimensionality reduction of target variable also should be done. Usually, it is done by alignment between independent and target variables. Therefore, for BCI data, the widely used dimensionality reduction algorithm is partial least squares (PLS) and its modifications [7], [4], [24]. The algorithm maps the features and targets to the shared latent space by maximizing the covariance between the projected vectors. This grants the capacity to store knowledge of the initial input and target matrices and discover their correlations. The latent space has a much smaller dimensionality than the original data description. It produces a stable linear model build on a reduced number of features.

For the data in tensor format, there are several modifications of this algorithm, such as High-Order PLS [23], Multi-linear PLS [24] and Multi-way PLS [4]. Another similar approach to PLS is a canonical-correlation analysis (CCA). It maximizes the correlations between projected vectors of the features and the targets.

In recent years, there has been an increasing amount of literature on deep learning (DL) methods for BCI task. The most recent DL models for BCI task

are Deep Tensor Canonical Correlation Analysis (DTCCA) [21] and Hybrid Autoencoder [16]. Shallow autoencoders are essentially equivalent to PCA transformations, especially when they are trained, using weight decay regularization. But autoencoders with nonlinear encoder functions and nonlinear decoder functions can learn a more powerful nonlinear generalization of PCA. Furthermore, if a condition module is added to an autoencoder for alignment with the target data, autoencoder's reduced features are connected with target data. Collectively, these studies outline a critical role for dimensionality reduction and that DL models become more prevalent in BCI task.

Despite the redundancy of data in the BCI problem, tensorization is sometimes used, which can help to catch hidden correlations and create additional informative features. Tensorization by appending an additional dimension as degrees of polynomial fittings helped to improve classification metrics [15]. Time delay embedding [5] is used to improve classification quality for BCI task. This approach was inspired by the common spectral patterns method [11] and allowed to identify bases in which the data differ substantially between classes. Moreover, for the task of artefact removal, generation of high--order tensor data for EEG was used [25]. This tensorization was made by applying wavelet transform. But none of these tensorizations has been applied in the task of predicting target time series.

One of the common tensorization techniques for time series is a hankelization. It is a natural data augmentation technique for time series to incorporate the intrinsic temporal correlation. For example, hankelization was used for anomaly detection in traffic data [20]. It has been shown that with the hankelization operation, the model can simultaneously capture the global and local spatio-temporal correlations and exhibit more robust performance. In addition, hankelization was used for forecasting for different time series data. For instance, forecasting of short term wind speed data was conducted via selective hankelization [9].

To apply hankelization for tensor data, multi-way delay embedding transform (MDT) was introduced [22]. It helped to capture some shift-invariant structure in the task of recovery of missing data. Moreover, MDT has been used to forecast short time series [18] and to forecast power consumption

[17]. In both works, its ability to capture hidden correlations increased the quality of forecasting.

In this work, the influence of hankelization on the quality of prediction was studied. We considered hankelization with such dimensionality reduction models as MPCA [13], HOPLS [23] and state-of-the-art deep learning model (autoencoder) [16]. The prediction was made for three samples of data: two from the Neurotycho dataset [2] and one from the human EEG dataset [8]. The results showed that hankelization can help to improve prediction quality by using it before different dimensionality reduction models.

2 Problem statement

In this work, we work with multiview data that can be presented in tensor format. So, in this chapter, the main tensor notations [3] are given and the task of decoding the time series is described. Moreover, the idea proposed in this work is described schematically. The idea is to predict target by combination of the tensorization method and dimensionality reduction models.

2.1 Notation

Tensors are multi-dimensional generalizations of matrices. The tensor is denoted by bold underlined capital letters, e.g., $\underline{\mathbf{X}} \in \mathbb{R}^{I_1 \times I_2 \times \ldots \times I_N}$. Matrices are denoted by boldface capital letters, e.g., $\mathbf{X} \in \mathbb{R}^{I_1 \times I_2}$, and vectors are denodet by boldface lower case letters, e.g., $\mathbf{x} \in \mathbb{R}^{I_1}$.

The element of tensor $\underline{\mathbf{X}}$ at position $(i_1, \ldots, i_N)$ will be denoted as $x_{i_1,\ldots,i_N} = \underline{\mathbf{X}}(i_1, \ldots, i_N) \in \mathbb{R}$. The order of a tensor is the number of dimensions, also known as "modes", "ways" or "dimensions".

One can define the inner product of two tensors $\underline{\mathbf{X}}$ and $\underline{\mathbf{Y}}$ as follows:

$$\langle \underline{\mathbf{X}}, \underline{\mathbf{Y}} \rangle = \sum_{i_1,i_2,\ldots,i_N} \underline{\mathbf{X}}(i_1, i_2, \ldots, i_N)\underline{\mathbf{Y}}(i_1, i_2, \ldots, i_N) = \sum_{i_1,i_2,\ldots,i_N} x_{i_1,\ i_2,\ldots,i_N} y_{i_1,\ i_2,\ldots,i_N}.$$

The norm of the tensor is defined by the inner product

$$\|\underline{\mathbf{X}}\| = \langle \underline{\mathbf{X}}, \underline{\mathbf{X}} \rangle^{1/2} = \sqrt{\sum_{i_1,i_2,\ldots,i_N} x^2_{i_1,\ i_2,\ldots,i_N}}.$$

The mode-n matricization of the tensor $\underline{\mathbf{X}} \in \mathbb{R}^{I_1 \times I_2 \times \ldots \times I_N}$ is defined as the matrix

$$\mathbf{X}_{(n)} \in \mathbb{R}^{I_n \times I_1 \cdot I_2 \cdot \ldots \cdot I_{n-1} \cdot I_{n+1} \cdot \ldots \cdot I_N},$$

with I_n rows and $I_1 \cdot I_2 \cdot \ldots \cdot I_{n-1} \cdot I_{n+1} \cdot \ldots \cdot I_N$ columns. It is also possible to vectorize a tensor $\underline{\mathbf{X}}$. The vectorized version is

$$\text{vec}(\underline{\mathbf{X}}) \in \mathbb{R}^{I_1 \cdot I_2 \cdot \ldots \cdot I_N}.$$

The mode-n (multilinear) product of a tensor $\underline{\mathbf{X}} \in \mathbb{R}^{I_1 \times I_2 \times \ldots \times I_N}$ with a matrix $\mathbf{U} \in \mathbb{R}^{J \times I_n}$ is denoted by

$$\underline{\mathbf{Y}} = \underline{\mathbf{X}} \times_n \mathbf{U} \in \mathbb{R}^{I_1 \times \ldots \times I_{n-1} \times J \times I_{n+1} \times \ldots \times I_N}.$$

Elementwise, we have

$$c_{i_1, \ldots, \, i_{n-1}, \, j, \, i_{n+1}, \ldots, \, i_N} = \sum_{i_n} x_{i_1, \ldots, \, i_n, \ldots, \, i_N} u_{j i_n}.$$

The equivalent matrix form is $\mathbf{C}_{(n)} = \mathbf{B}\mathbf{A}_{(n)}$. The mode-$n$ product of a tensor $\underline{\mathbf{X}}$ with a matrix $\mathbf{U}$ is related to a change of basis in the case when a tensor defines a multilinear operator.

A full multilinear product of an Nth-order tensor, $\underline{\mathbf{X}} \in \mathbb{R}^{I_1 \times I_2 \times \ldots \times I_N}$, and a set of N factor matrices, $\mathbf{U}^{(n)} \in \mathbb{R}^{I_n \times R_n}$ for $n = 1, 2, \ldots, N$ can be compactly written as

$$\underline{\mathbf{Y}} = \underline{\mathbf{X}} \times_1 \mathbf{U}^{(1)} \times_2 \mathbf{U}^{(2)} \ldots \times_N \mathbf{U}^{(N)} = [\![\underline{\mathbf{X}}; \, \mathbf{U}^{(1)}, \mathbf{U}^{(2)}, \ldots, \mathbf{U}^{(N)}]\!] \in \mathbb{R}^{R_1 \times R_2 \times \ldots \times R_N}.$$

It provides a factorization of a Nth-order tensor into a relatively small size core tensor and factor matrices. It can be written as follows

$$\underline{\mathbf{X}} \approx [\![\underline{\mathbf{G}}; \, \mathbf{U}^{(1)}, \mathbf{U}^{(2)}, \ldots, \mathbf{U}^{(N)}]\!],$$

where $\underline{\mathbf{X}} \in \mathbb{R}^{I_1 \times I_2 \times \ldots \times I_N}$ is the given tensor, $\underline{\mathbf{G}} \in \mathbb{R}^{R_1 \times R_2 \times \ldots \times R_N}$ is the core tensor, $R_n < I_n, n = 1, \ldots, N$, and $\mathbf{U}^{(n)} \in \mathbb{R}^{I_n \times R_n}$, $n = 1, \ldots, N$ are the factor matrices. In the standart Tucker model, factor matrices are orthogonal.

2.2 Problem statement

Definition 2.1. A time series is a function of a discrete argument $\mathbf{s}(t)$ that matches time reports $t_i \in \mathcal{T}$ with a vector of the values of the measured variables $\mathbf{s}(t_i) = s_i \in \mathbb{R}^M$.

Definition 2.2. Let $\{\mathbf{s}_n(t)\}_{n=1}^{N_s}$ be a set of time series and $\{\mathbf{y}_n(t)\}_{n=1}^{N_y}$ a set of target time series. The task of finding the values of $\{\mathbf{y}_n(t)\}_{n=1}^{N_y}$ from the previous values of $\{\mathbf{s}_n(t)\}_{n=1}^{N_s}$ is called the task of decoding time series $\{\mathbf{y}_n(t)\}_{n=1}^{N_y}$.

A multiview dataset made from the time series $\mathbf{s}(t),\ \mathbf{y}(t)$:

$$\underline{\mathbf{X}} \in \mathbb{R}^{M \times I_1 \times \cdots \times I_{D_x}}, \qquad \underline{\mathbf{Y}} \in \mathbb{R}^{M \times J_1 \times \cdots \times J_{D_y}}, \tag{2.2.1}$$

where $\mathbf{y}_m = \mathbf{y}(t_m)$, $\underline{\mathrm{X}}_m \in \mathbb{R}^{I_1 \times \cdots \times I_{D_x}}$ is a tensor.

The goal is to forecast a dependent variable $\mathbf{y}_m$ from an independent input object $\underline{\mathrm{X}}_m$, $m = 1, \ldots, M$.

In this research, the aim is to find a composition of models such as:

$$\Phi = \Psi_x \circ h_x \circ g \circ h_y^{-1} \circ \Psi_y^{-1} : \quad \mathbb{R}^{I_1 \times \cdots \times I_{D_x}} \to \mathbb{R}^{J_1 \times \cdots \times J_{D_y}}, \tag{2.2.2}$$

where Ψ_x, h_x, g, h_y and Ψ_x work within a process:

$$\begin{array}{ccc}
\underline{\mathbf{X}} & \xrightarrow{\Phi} & \underline{\mathbf{Y}} \\
{\scriptstyle \Psi_x}\downarrow & & \downarrow{\scriptstyle \Psi_y} \\
\hat{\underline{\mathbf{X}}} & \xrightarrow{\underline{\Phi}} & \hat{\underline{\mathbf{Y}}} \\
{\scriptstyle h_x}\downarrow & & \downarrow{\scriptstyle h_y} \\
\underline{\mathbf{T}} & \xrightarrow[g]{} & \underline{\mathbf{U}}
\end{array}$$

Definition 2.3. Tensorization model $\Psi : \mathbb{R}^{M \times I_1 \times \cdots \times I_D} \to \mathbb{R}^{\tilde{M} \times \tilde{I}_1 \times \cdots \times \tilde{I}_G}$, where $D < G$, is a model that increase order of tensor from D to G.

In this research, hankelization (Sec. 3.1) is used as the tensorization model. In case without tensorization, Ψ is the identity transformation. There are four types of the tensorization model used:

- Without tensorization;
- Hankelization along time dimension;
- Hankelization along space dimension;
- Hankelization along time and space dimensions.

The type of tensorization ζ is a hyperparameter of the model $\Phi(\zeta)$.

Definition 2.4. $h : \mathbb{R}^{\tilde{I}_1 \times \cdots \times \tilde{I}_G} \to \mathbb{R}^{L_1 \times \ldots \times L_G}$, where $L_i < I_i$ is called a dimensionality reduction model, which reduces the dimensionality from $\mathbb{R}^{\tilde{I}_1 \times \cdots \times \tilde{I}_G}$ to $\mathbb{R}^{L_1 \times \ldots \times L_G}$, namely $\underline{\mathbf{T}}_m = h_x\big(\underline{\mathbf{X}}_m, \theta_y\big)$ where θ are parameters of the model.

Definition 2.5. $g : \mathbb{R}^{L_1 \times \ldots \times L_{G_x}} \to \mathbb{R}^{K_1 \times \ldots \times K_{G_y}}$ is called an alignment model, which aligns each $\underline{\mathbf{T}}_m \in \mathbb{R}^{L_1 \times \ldots \times L_{G_x}}$ to $\underline{\mathbf{U}}_m \in \mathbb{R}^{K_1 \times \ldots \times K_{G_y}}$, namely $\underline{\mathbf{U}}_m = g\big(\underline{\mathbf{T}}_m, \eta\big)$, where η are parameters of the model.

Model Φ is called optimal, if it minimizes error functional $\mathcal{L}$:

$$\Phi^* = \underset{\{\theta,\eta,\zeta\}}{\arg\min} \mathcal{L}\left(\Phi(\underline{\mathbf{X}}, \theta, \eta, \zeta),\ \underline{\mathbf{Y}}\right), \tag{2.2.3}$$

where θ is the parameters of a dimensionality reduction model, η is the parameters of an alignment model and ζ is a type of tensorization.

In this work, we investigate how tensorization type, in combination with different dimensionality models, influences on the quality of the prediction.

Method for tensorization, hankelization, is described in the Section 3.1. Possible models of dimensionality reduction are discussed in the Sections 3.2-??. Tensor regression that can be used as an alignment model is described in the Section 3.7.

3 Methodology

There are a number of instruments available for dimensionality reduction, such as principal component analysis (PCA), partial least squares (PLS), their tensor versions (High-Order PLS, MPCA) and autoencoder. These methods were used directly for the initial data or for data tensorized by its structure. In this research, hankelization was used as a tensorization method before dimensionality reduction techniques. Hankelization is one of the most common tensorization methods for time series. The benefit of this approach is that it reveals hidden correlations in initial data and allows to find low-rank approximations with high levels of compression. In this chapter, hankelization method, dimensionality reduction, and alignment models are described.

3.1 Hankelization

To begin, we will provide a standard definition of a hankelization for a vector that can be understood as a time-series signal. Let consider $\mathbf{x} = (x_1, \ldots, x_I)^\top \in \mathbb{R}^I$, a hankelization of $\mathbf{x}$ with a parameter τ is given by

$$\mathcal{H}_\tau(\mathbf{x}) = \begin{pmatrix} x_1 & x_2 & \ldots & x_{I-\tau+1} \\ x_2 & x_3 & \ldots & x_{I-\tau+2} \\ \vdots & \vdots & \ddots & \vdots \\ x_\tau & x_{\tau+1} & \ldots & x_I \end{pmatrix} \in \mathbb{R}^{\tau \times (I-\tau+1)},$$

where $\mathcal{H}_\tau(\mathbf{x})$ is a Hankel matrix of a vector $\mathbf{x}$.

Let consider a 1D convolution for vector $x \in \mathbb{R}^I$ and kernel $h \in \mathbb{R}^k$, then 1D convolution is represented by

$$x \ * \ h = \begin{pmatrix} x_1 & x_2 & \ldots & x_k \\ x_2 & x_3 & \ldots & x_{k+1} \\ \vdots & \vdots & \ddots & \vdots \\ x_{I-k+1} & x_{I-k+2} & \ldots & x_I \end{pmatrix} \begin{pmatrix} h_1 \\ h_2 \\ \vdots \\ h_k \end{pmatrix} = \mathcal{H}_{I-k+1}(x)h.$$

Time-series have variables that are temporally nearby are highly correlated. In the BCI task, signals from different channels are also correlated. So,

the data in BCI are spatially and temporally correlated. Local correlations are the reasons for the well-known advantages of extracting and combining local features before making a prediction. Convolution extracts local features by restricting the receptive fields of hidden units to be local [1]. The above mentioned connection of hankelization with convolution makes hankelization the helpful tool for revealing the hidden correlations.

$$\mathbf{S} = \begin{bmatrix} \begin{matrix} 1 & & 0 \\ & \ddots & \\ 0 & & 1 \end{matrix} & \tau & 0 \\ & \begin{matrix} 1 & & 0 \\ & \ddots & \\ 0 & & 1 \end{matrix} & \\ & & \begin{matrix} 1 & & 0 \\ & \ddots & \\ 0 & & 1 \end{matrix} \\ & & & \begin{matrix} 1 & & 0 \\ & \ddots & \\ 0 & & 1 \end{matrix} \\ 0 & & & \begin{matrix} 1 & & 0 \\ & \ddots & \\ 0 & & 1 \end{matrix} \end{bmatrix} \tag{3.1.1}$$

Hankelization is an effective way to transform lower-order data to higher-order tensors. MDT is a multi-way extension of hankelization [22]. It combines multi-linear duplication and multi-way folding operations. By denoting $\underline{\hat{\mathbf{X}}} \in \mathbb{R}^{\tilde{I}_1 \times \ldots \times \tilde{I}_G}$ as the block Hankel tensor of $\underline{\mathbf{X}} \in \mathbb{R}^{I_1 \times \ldots \times I_D}$, $D < G$, the MDT for $\underline{\mathbf{X}}$ is defined by

$$\underline{\hat{\mathbf{X}}} = \mathcal{H}_\tau(\underline{\mathbf{X}}) = \text{Fold}_{(\mathbf{I},\, \tau)} \left(\underline{\mathbf{X}} \times_1 \mathbf{S}_1 \ldots \times_D \mathbf{S}_D\right),$$

where $\mathbf{S}_d \in \mathbb{R}^{\tau_d(I_d - \tau_d + 1) \times I_d}, d = 1, \ldots, D$ is a duplication matrix (3.1.1) and

$$\text{Fold}_{(I,\, \tau)} : \mathbb{R}^{\tau_1(I_1 - \tau_1 + 1) \times \ldots \times \tau_D(I_D - \tau_D + 1)} \to \mathbb{R}^{\tau_1 \times (I_1 - \tau_1 + 1) \times \ldots \times \tau_D \times (I_D - \tau_D + 1)}$$

constructs a higher order block Hankel tensor $\underline{\hat{\mathbf{X}}}$ from the input tensor $\underline{\mathbf{X}}$. The inverse MDT for $\underline{\hat{\mathbf{X}}}$ is given by

$$\underline{\mathbf{X}} = \mathcal{H}_\tau^{-1}(\underline{\hat{\mathbf{X}}}) = \text{Unfold}_{(\mathbf{I},\, \tau)}(\underline{\hat{\mathbf{X}}}) \times_1 \mathbf{S}_1^\dagger \ldots \times_N \mathbf{S}_N^\dagger,$$

where $\dagger$ is the Moore-Penrose pseudo-inverse.

3.2 MPCA

Principal Component Analysis (PCA) [10] is defined for a matrix $\mathbf{X} \in \mathbb{R}^{M \times I}$. PCA seek a linear combination of the columns of matrix $\mathbf{X}$ with maximum variance:

$$\mathbf{X}\mathbf{a} = \sum_i a_i \mathbf{x}_i.$$

The variance of any such linear combination is given by

$$\mathbf{a}^\top \mathbf{S}\mathbf{a} = \mathrm{var}\left(\mathbf{X}\mathbf{a}\right),$$

where $\mathbf{S}$ is the sample covariance matrix associated with the dataset. Identifying the linear combination with maximum variance is equivalent to obtaining a p-dimensional vector $\mathbf{a}$ which maximizes the quadratic form $\mathbf{a}^\top \mathbf{S}\mathbf{a}$. For this problem to have a well-defined solution

$$\mathbf{S}\mathbf{a} = \lambda \mathbf{a},$$

where $\mathbf{a}$ must be a (unit-norm) eigenvector, and λ the corresponding eigenvalue, of the covariance matrix $\mathbf{S}$.

Any $I \times I$ real symmetric matrix, such as a covariance matrix $\mathbf{S}$, has exactly I real eigenvalues, λ_i $(i = 1, \ldots, I)$, and their corresponding eigenvectors can be defined to form an orthonormal set of vectors, i.e. $\mathbf{a}_j^\top \mathbf{a}_i = 1$. The linear combinations such as $\mathbf{X}\mathbf{a}_i$ $(i = 1, \ldots, I)$ are called the principal components of the dataset.

For multiview data, Multilinear Principal Component Analysis (MPCA) is used. It performs feature extraction by determining a multilinear projection that captures most of the original tensorial input variation. MPCA objective is to define a multilinear transformation $\left\{\tilde{\mathbf{U}}^{(d)} \in \mathbb{R}^{I_d \times L_d},\ d = 1, \ldots, D\right\}$ that maps the original tensor space $\mathbb{R}^{I_1} \otimes \mathbb{R}^{I_2} \otimes \ldots \otimes \mathbb{R}^{I_D}$ into a tensor subspace $\mathbb{R}^{L_1} \otimes \mathbb{R}^{L_2} \otimes \ldots \otimes \mathbb{R}^{L_D}$ (with $L_d < I_d$, $d = 1, \ldots, D$):

$$\hat{\underline{\mathbf{X}}}_m \approx \underline{\mathbf{X}}_m \times_1 \tilde{\mathbf{U}}^{(1)\top} \times_2 \tilde{\mathbf{U}}^{(2)\top} \ldots \times_D \tilde{\mathbf{U}}^{(D)\top}, \quad m = 1, \ldots, M,$$

such that $\left\{\hat{\underline{\mathbf{X}}}_m \in \mathbb{R}^{L_1} \otimes \mathbb{R}^{L_2} \otimes \ldots \otimes \mathbb{R}^{L_D}, \quad m = 1, \ldots, M\right\}$ captures most of the variations observed in the original tensor objects, assuming that these variations are measured by the total tensor scatter $\Upsilon_{\underline{\mathbf{X}}}$ which can be calculated as follows:

$$\Upsilon_{\underline{\mathbf{X}}} = \sum_{m=1}^{M} \|\underline{\mathbf{X}}_m - \overline{\underline{\mathbf{X}}}\|_F^2,$$

where $\overline{\underline{\mathbf{X}}} = \frac{1}{M}\sum_{m=1}^{M} \underline{\mathbf{X}}_m$

The MPCA objective is the determination of the D projection matrices $\left\{\tilde{\mathbf{U}}^{(d)} \in \mathbb{R}^{I_d \times L_d},\ d = 1, \ldots, D\right\}$ that maximize the total tensor scatter $\Upsilon_{\underline{\mathbf{X}}}$

$$\left\{\tilde{\mathbf{U}}^{(d)} \in \mathbb{R}^{I_d \times L_d},\ d = 1, \ldots, D\right\} = \arg \max_{\tilde{\mathbf{U}}^{(1)}, \ldots, \tilde{\mathbf{U}}^{(d)}} \Upsilon_{\underline{\mathbf{X}}}.$$

Here, the dimensionality L_d for each mode is assumed to be known or predetermined.

3.3 Partial Least Squares

In BCI area, data usually have not only multiple measurements of brain signals but also multiple measurements of target time series. So, target variables have correlations. Therefore, the dimensionality of target variables should be reduced too. We can use the partial least squares (PLS) algorithm that reduce dimensionality by alignment between target variables and features [6], [7]. PLS projects the design matrix $\mathbf{X}$ and the target matrix $\mathbf{Y}$ to the latent space with low dimensionality ($l < n$). The PLS algorithm finds the latent space matrices $\mathbf{T}, \mathbf{U} \in \mathbb{R}^{m \times l}$ that best describe the original matrices $\mathbf{X}$ and $\mathbf{Y}$.

The design matrix $\mathbf{X}$ and the target matrix $\mathbf{Y}$ are projected into the latent space in the following way:

$$\underset{m\times n}{\mathbf{X}} = \underset{m\times l}{\mathbf{T}} \cdot \underset{l\times n}{\mathbf{P}^{\mathsf{T}}} + \underset{m\times n}{\mathbf{F}} = \sum_{k=1}^{l} \underset{m\times 1}{\mathbf{t}_k} \cdot \underset{1\times n}{\mathbf{p}_k^{\mathsf{T}}} + \underset{m\times n}{\mathbf{F}}, \tag{3.3.1}$$

$$\underset{m\times r}{\mathbf{Y}} = \underset{m\times l}{\mathbf{U}} \cdot \underset{l\times r}{\mathbf{Q}^{\mathsf{T}}} + \underset{m\times r}{\mathbf{E}} = \sum_{k=1}^{l} \underset{m\times 1}{\mathbf{u}_k} \cdot \underset{1\times r}{\mathbf{q}_k^{\mathsf{T}}} + \underset{m\times r}{\mathbf{E}}, \tag{3.3.2}$$

where $\mathbf{T}$ and $\mathbf{U}$ are the scores matrices in the latent space, $\mathbf{P}$ and $\mathbf{Q}$ are the loading matrices, $\mathbf{E}$ and $\mathbf{F}$ are residual matrices. The PLS maximizes the linear relation between the columns of matrices $\mathbf{T}$ and $\mathbf{U}$ as

$$\mathbf{U} \approx \mathbf{TB}, \quad \mathbf{B} = \text{diag}(\beta_k), \quad \beta_k = \mathbf{u}_k^{\mathsf{T}}\mathbf{t}_k/(\mathbf{t}_k^{\mathsf{T}}\mathbf{t}_k).$$

We use the PLS algorithm as the dimensionality reduction algorithm in this research.

To obtain the model prediction and find the model parameters, we multiply both sides of (3.3.1) by $\mathbf{W}$. Since the residual matrix $\mathbf{E}$ rows are orthogonal to the columns of $\mathbf{W}$, we have

$$\mathbf{XW} = \mathbf{TP}^{\mathsf{T}}\mathbf{W}.$$

The linear transformation between objects in the input and latent spaces is the following

$$\mathbf{T} = \mathbf{XW}^*, \quad \text{where } \mathbf{W}^* = \mathbf{W}(\mathbf{P}^{\mathsf{T}}\mathbf{W})^{-1}. \tag{3.3.3}$$

The matrix of the model parameters $\mathbf{\Theta}$ could be found from the equations (3.3.2) and (3.3.3) as

$$\mathbf{Y} = \mathbf{UQ}^{\mathsf{T}} + \mathbf{E} \approx \mathbf{TBQ}^{\mathsf{T}} + \mathbf{E} = \mathbf{XW}^*\mathbf{BQ}^{\mathsf{T}} + \mathbf{E} = \mathbf{X\Theta} + \mathbf{E}. \tag{3.3.4}$$

Thus, the model parameters (3.6.2) are equal to

$$\mathbf{\Theta} = \mathbf{W}(\mathbf{P}^{\mathsf{T}}\mathbf{W})^{-1}\mathbf{BQ}^{\mathsf{T}}.$$

The final model (3.3.4) is a linear model that is low-dimensional in the latent space. It reduces the data redundancy and increases the model stability.

3.4 High-Order PLS

HOPLS [23], [3] performs simultaneously constrained Tucker decompositions for an $(D+1)$th-order independent tensor, $\underline{\mathbf{X}} \in \mathbb{R}^{M \times I_1 \times \ldots \times I_D}$, and an $(D+1)$th-order dependent tensor, $\underline{\mathbf{Y}} \in \mathbb{R}^{M \times J_1 \times \ldots \times J_D}$, which have the same size in the first mode. Such a model allows us to find the optimal subspace

approximation of $\underline{\mathbf{X}}$, in which the independent and dependent variables share a common set of latent vectors in one specific mode (i.e.,samples mode).

$$\underline{\mathbf{X}} = \sum_{r=1}^{R} \underline{\mathbf{G}}_{xr} \times_1 \mathbf{t}_r \times_2 \mathbf{P}_r^{(1)} \ldots \times_{D+1} \mathbf{P}_r^{(D)} + \underline{\mathbf{E}}_R,$$
$$\underline{\mathbf{Y}} = \sum_{r=1}^{R} \underline{\mathbf{G}}_{yr} \times_1 \mathbf{t}_r \times_2 \mathbf{Q}_r^{(1)} \ldots \times_{D+1} \mathbf{Q}_r^{(D)} + \underline{\mathbf{F}}_R,$$

where R is the number of latent vectors, $\mathbf{t}_r \in \mathbb{R}^M$ is the r-th latent vector, $\left\{\mathbf{P}_r^{(d)}\right\}_{d=1}^{D} \in \mathbb{R}^{I_d \times L_d}$ and $\left\{\mathbf{Q}_r^{(d)}\right\}_{d=1}^{D} \in \mathbb{R}^{J_d \times K_d}$ are the loading matrices in mode-d, and $\underline{\mathbf{G}}_{xr} \in \mathbb{R}^{1 \times L_1 \times \ldots \times L_D}$ and $\underline{\mathbf{G}}_{yr} \in \mathbb{R}^{1 \times K_1 \times \ldots \times K_D}$ are core tensors.

By defining a latent matrix $\mathbf{T} = \left[\mathbf{t}_1, \ldots, \mathbf{t}_R\right]$, mode-$d$ loading matrix $\overline{\mathbf{P}}^{(d)} = \left[\mathbf{P}_1^{(d)}, \ldots, \mathbf{P}_R^{(d)}\right]$, mode-$d$ loading matrix $\overline{\mathbf{Q}}^{(d)} = \left[\mathbf{Q}_1^{(d)}, \ldots, \mathbf{Q}_R^{(d)}\right]$ and core tensors

$$\underline{\mathbf{G}}_x = \text{blockdiag}\left(\underline{\mathbf{G}}_{x1}, \ldots, \underline{\mathbf{G}}_{xR}\right) \in \mathbb{R}^{R \times RL_1 \times \ldots \times RL_D},$$
$$\underline{\mathbf{G}}_y = \text{blockdiag}\left(\underline{\mathbf{G}}_{y1}, \ldots, \underline{\mathbf{G}}_{yR}\right) \in \mathbb{R}^{R \times RK_1 \times \ldots \times RK_D},$$

the HOPLS model can be rewritten as

$$\underline{\mathbf{X}} = \underline{\mathbf{G}}_x \times_1 \mathbf{T} \times_2 \overline{\mathbf{P}}^{(1)} \ldots \times_{D+1} \overline{\mathbf{P}}^{(D)} + \underline{\mathbf{E}}_R,$$
$$\underline{\mathbf{Y}} = \underline{\mathbf{G}}_y \times_1 \mathbf{T} \times_2 \overline{\mathbf{Q}}^{(1)} \ldots \times_{D+1} \overline{\mathbf{Q}}^{(D)} + \underline{\mathbf{F}}_R,$$

where $\underline{\mathbf{E}}_R$ and $\underline{\mathbf{F}}_R$ are the residuals obtained after extracting R latent components. The core tensors, $\underline{\mathbf{G}}_x$ and $\underline{\mathbf{G}}_y$, have a special block-diagonal structure (Fig. 3.1) and their elements indicate the level of local interactions between the corresponding latent vectors and loading matrices.

The cross-covariance tensor is defined by

$$\underline{\mathbf{C}} = COV_{\{1,1\}}(\underline{\mathbf{X}}, \underline{\mathbf{Y}}) \in \mathbb{R}^{I_1 \times \ldots \times I_D \times J_1 \times \ldots \times J_D}$$

The optimization problem can be formulated as

$$\left\| [\![\underline{\mathbf{C}}; \mathbf{P}^{(1)\top}, \ldots, \mathbf{P}^{(D)\top}, \mathbf{Q}^{(1)\top}, \ldots, \mathbf{Q}^{(D)\top}]\!] \right\|_F^2 \rightarrow \max_{\substack{\left\{\mathbf{P}^{(d)}, \mathbf{Q}^{(d)}\right\}, \\ \text{s.t. } \mathbf{P}^{(d)\top}\mathbf{P}^{(d)} = \mathbf{I}_{L_d}, \\ \mathbf{Q}^{(d)\top}\mathbf{Q}^{(d)} = \mathbf{I}_{K_d}}},$$

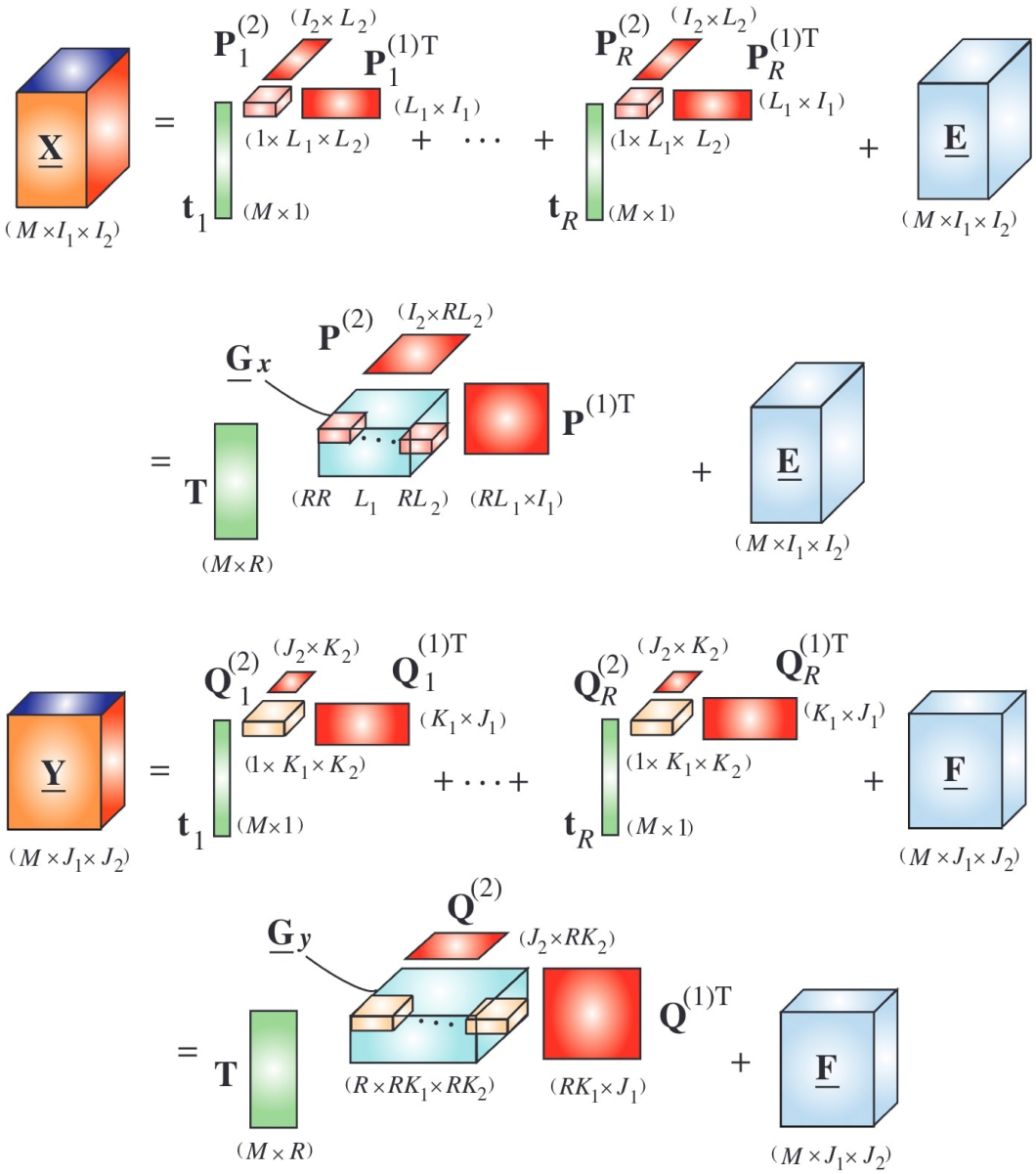


Figure 3.1. The HOPLS model which approximates the independent variables, $\underline{\mathbf{X}}$, as a sum of rank-$(1, L_1, L_2)$ tensors. The approximation for the dependent variables, $\underline{\mathbf{Y}}$, follows a similar principle, whereby the common latent components, $\mathbf{T}$, are shared between $\underline{\mathbf{X}}$ and $\underline{\mathbf{Y}}$. This picture is taken from [3].

where $[\![\ldots]\!]$ denotes the multilinear products between a tensor and a set of matrices, $\mathbf{P}^{(d)}$ and $\mathbf{Q}^{(d)}, d = 1, \ldots, D$, comprise the unknown parameters.

3.5 Autoencoder ReducedNet

Autoencoders are often used for the problem of dimensionality reduction. For example, for the task of reducing the dimension for electrocorticogram

data, the hybrid autoencoder ReducedNet [16] was used. Predicting the coordinates of hand movements is found to be more accurate after dimensionality reduction using ReducedNet than after methods such as PLS or KernelPCA. Therefore, it was decided to apply this model in this work as state-of-art model, but with minor modifications.

The model proposed in [16] consists of two blocks: a dimensionality reduction module consisting of an encoder h and a decoder ψ; and a condition module ω. The dimensionality reduction module is composed of convolutional blocks (Conv1D) and LSTM blocks. Each convolutional block contains of a one-dimensional convolutional layer, one Batch Normalization layer, GELU (Gaussian error linear unit) activation function layer and Dropout layer. An LSTM block is a single LSTM layer. The condition module was a single linear layer.

The ReducedNet model has only one tunable parameter, N_{CH}, that was the number of features after the encoder.

The loss, proposed in article [16], takes into account outputs from both modules:

$$\mathcal{L}_1 = \mathcal{L}_{rec} + \alpha \cdot \mathcal{L}_{dec}, \tag{3.5.1}$$

where reconstruction loss is defined as follows:

$$\mathcal{L}_{rec} = \frac{1}{M} \sum_{m=1}^{M} \|\underline{\mathbf{X}}_m - h \circ \psi(\underline{\mathbf{X}}_m)\|^2,$$

decoding loss is given by

$$\mathcal{L}_{dec} = \frac{1}{M} \sum_{m=1}^{M} \|\mathbf{Y}_m - \omega \circ h(\underline{\mathbf{X}}_m)\|^2,$$

and α is a parameter of loss function. We used $\alpha = 3$ as was used in the original paper [16].

The main problem of ReducedNet is that encoder's output is a matrix. It means that the input tensor loses its initial structure. Besides, it can work only with 3D data. Thus, in this work, TensorReducedNet is proposed. TensorReducedNet can be applyed to the data with any number of dimensions and it's encoders output is a tensor. The 3D example of TensorReducedNet

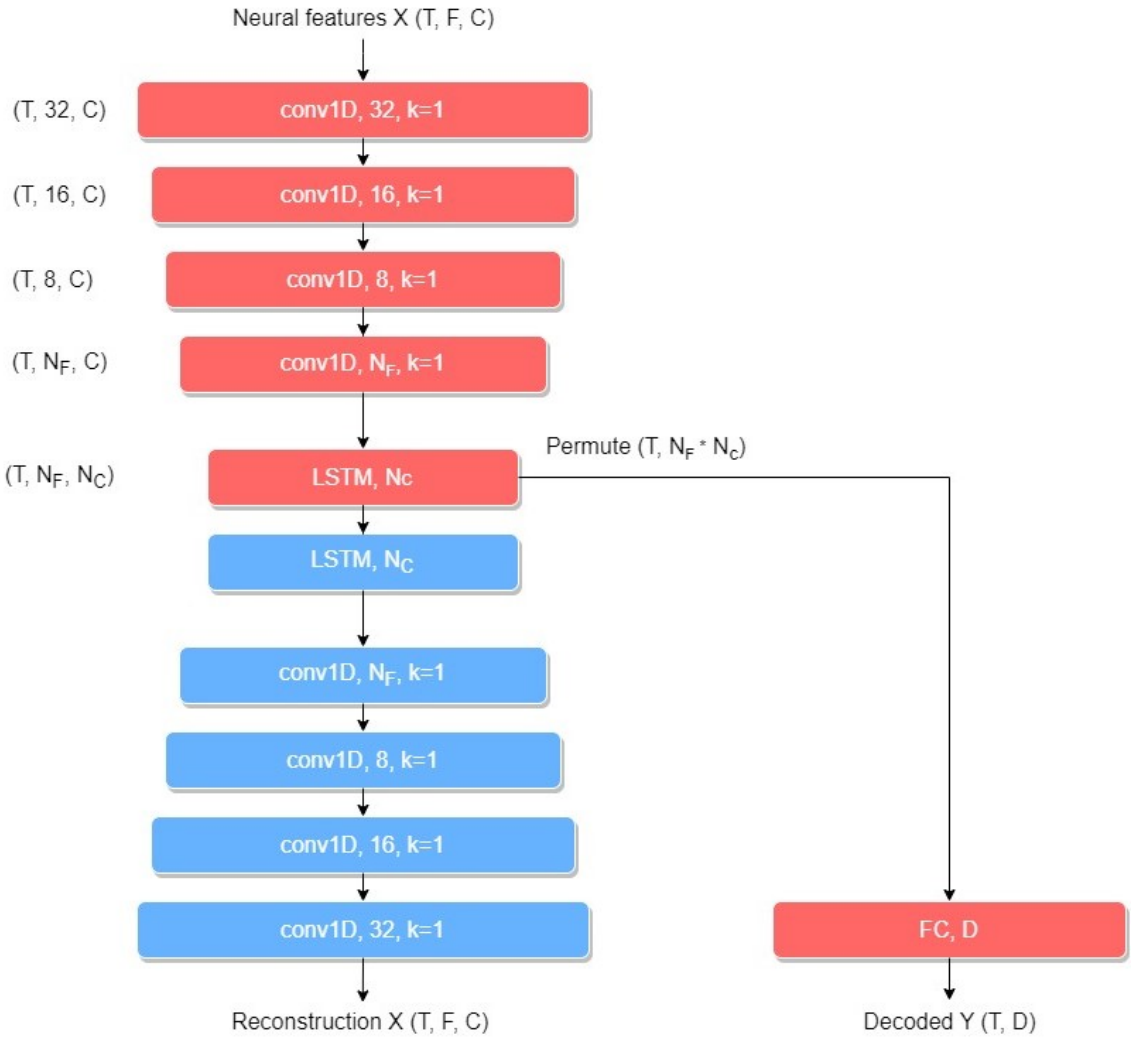


Figure 3.2. TensorReducedNet.

can be seen in Fig. 3.2. The output from the encoder of 3D TensorReducedNet is in $\mathbb{R}^{T \times N_F \times N_{CH}}$.

For tensors with order higher than 3, the reduction of dimensionality of all modes except one is achieved by convolutional blocks, the reduction of dimensionality of the last mode (temporal) is made by LSTM. For autoencoder with output in $\mathbb{R}^{T \times L_1 \times L_2 \times \ldots \times L_G}$ the changes done by encoder are the follows:

$$\underline{\mathbf{X}} \in \mathbb{R}^{M \times I_1 \times \ldots \times I_G} \xrightarrow{\text{conv1D blocks}} \underline{\tilde{\mathbf{X}}} \in \mathbb{R}^{M \times I_1 \times L_2 \times \ldots \times L_G} \xrightarrow{\text{LSTM block}} \underline{\hat{\mathbf{X}}} \in \mathbb{R}^{M \times L_1 \times L_2 \times \ldots \times L_G}.$$

The FC layer for TensorReducedNet is applied to the unfolded tensor $\underline{\hat{\mathbf{X}}}$ along first mode.

3.6 Multimodal regression

The feature tensor $\underline{\mathrm{X}}$ has $D+1$ dimension. To restore the target time series, the feature tensor $\underline{\mathrm{X}}$ can be unfolded by the first dimension:

$$\underline{\mathbf{X}}_{(1)} = \left[\mathrm{vec}\left(\underline{\mathbf{X}}_1\right)^\top, \ldots, \mathrm{vec}\left(\underline{\mathbf{X}}_M\right)^\top\right]^\top \in \mathbb{R}^{M\times(I_1\cdot\ldots\cdot I_D)} \tag{3.6.1}$$

Thus, the problem becomes a multimodal regression problem, where $\mathbf{X} \in \mathbb{R}^{M\times(I_1\cdot\ldots\cdot I_D)}$ is the input matrix obtained by matricization of the input tensor, $\mathbf{Y} \in \mathbb{R}^{M\times J}$ is the target matrix.

The purpose is to predict a dependent variable $\mathbf{y}_m \in \mathbb{R}^J$ with J targets from an independent variable $\mathbf{x}_m \in \mathbb{R}^I$ that has $I = I_1 \cdot \ldots \cdot I_D$ features. We hypothesize that there is a linear correlation between the object $\mathbf{x}_m$ and the predicted variable $\mathbf{y}_m$ as

$$\mathbf{y}_m = \boldsymbol{\Theta}\mathbf{x}_m + \boldsymbol{\varepsilon}, \tag{3.6.2}$$

where $\boldsymbol{\Theta} \in \mathbb{R}^{J\times I}$ is the matrix of the model parameters, and $\boldsymbol{\varepsilon} \in \mathbb{R}^J$ is a residual vector. The matrix of the model parameters $\boldsymbol{\Theta}$ can be found from a provided dataset $(\mathbf{X}, \mathbf{Y})$ where $\mathbf{X} \in \mathbb{R}^{M\times I}$ is the matrix of design and $\mathbf{Y} \in \mathbb{R}^{M\times J}$ is the target matrix:

$$\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_m]^\mathsf{T} = [\boldsymbol{\chi}_1, \ldots, \boldsymbol{\chi}_n]; \quad \mathbf{Y} = [\mathbf{y}_1, \ldots, \mathbf{y}_m]^\mathsf{T} = [\boldsymbol{\nu}_1, \ldots, \boldsymbol{\nu}_r].$$

The columns $\boldsymbol{\chi}_j$ of $\mathbf{X}$ correspond to the object features, and the columns $\boldsymbol{\nu}_j$ of $\mathbf{Y}$ correspond to the targets.

The optimal parameters are obtained by the reduction of an error function. The quadratic loss function can be defined as follows:

$$\mathcal{L}(\boldsymbol{\Theta}|\mathbf{X}, \mathbf{Y}) = \left\| \underset{M\times J}{\mathbf{Y}} - \underset{M\times I}{\mathbf{X}} \cdot \underset{J\times I}{\boldsymbol{\Theta}}^\mathsf{T} \right\|_2^2 \to \min_{\boldsymbol{\Theta}}. \tag{3.6.3}$$

The solution of (3.6.3) is given by

$$\boldsymbol{\Theta} = \mathbf{Y}^\mathsf{T}\mathbf{X}(\mathbf{X}^\mathsf{T}\mathbf{X})^{-1}.$$

The linear dependency of the columns of $\mathbf{X}$ leads to an unstable solution for the optimization problem (3.6.3). If there is a vector $\boldsymbol{\alpha} \neq \mathbf{0}_I$ such that $\mathbf{X}\boldsymbol{\alpha} = \mathbf{0}_m$, then adding $\boldsymbol{\alpha}$ to any column of $\boldsymbol{\Theta}$ does not change the value

of the loss function $\mathcal{L}(\mathbf{\Theta}|\mathbf{X}, \mathbf{Y})$. In this case, the matrix $\mathbf{X}^\mathsf{T}\mathbf{X}$ is close to singular and is not invertible. One of the possible solutions is not to make a matricization of the input tensor but to use tensor regression. The main advantage of the tensor regression over the multimodal regression is that it stores information about the data structure.

3.7 Tensor regression

The tensor regression can be defined in a similar way as (3.6.2):

$$\mathbf{y}_m = \langle \underline{\mathbf{X}}_m | \underline{\mathbf{W}} \rangle + \varepsilon, \tag{3.7.1}$$

where $\langle \underline{\mathbf{X}}_m | \underline{\mathbf{W}} \rangle$ is a tensor contraction along the first D dimensions of the D-dimensional initial tensor $\underline{\mathbf{X}}_m \in \mathbb{R}^{I_1 \times \cdots \times I_D}$ and $(D+1)$-dimensional tensor of model parameters $\underline{\mathbf{W}} \in \mathbb{R}^{I_1 \times \cdots \times I_D \times J}$, $\varepsilon \in \mathbb{R}^J$ is an error, $\mathbf{y}_m \in \mathbb{R}^J$ is the target vector.

The j-th element of the vector obtained after tensor contraction of $\underline{\mathbf{X}}_m$ and $\underline{\mathbf{W}}$ is calculated as follows:

$$\langle \underline{\mathbf{X}}_m | \underline{\mathbf{W}} \rangle_j = \sum_{i_1=1}^{I_1} \cdots \sum_{i_D=1}^{I_D} x_{i_1,\ldots,i_D} w_{i_1,\ldots,i_D,j} \tag{3.7.2}$$

The optimal parameter tensor $\underline{\mathbf{W}}^*$ is found by minimizing the quadratic error function:

$$\underline{\mathbf{W}}^* = \arg\min_{\underline{\mathbf{W}}} \sum_{m=1}^{M} \left\| \mathbf{y}_m - \langle \underline{\mathbf{X}}_m | \underline{\mathbf{W}} \rangle \right\|_2^2 \tag{3.7.3}$$

In practice, the parameter tensor $\underline{\mathbf{W}}$ is represented in the Tucker decomposition:

$$\underline{\mathbf{W}} \approx \underline{\mathbf{G}} \times_1 \mathbf{U}^{(1)} \ldots \times_D \mathbf{U}^{(D)} \times_{D+1} \mathbf{U}^{(D+1)}, \tag{3.7.4}$$

where $\underline{\mathbf{G}}$ is a smaller core tensor, $\mathbf{U}^{(i)}$ are unitary matrices.

We used the Tucker regression model as it has several advantages. One of them is an improved modeling capability that is more parsimonious and compact, particularly useful when working with a limited number of available samples. Also, stable algorithms based on SVD decomposition exist for the Tucker decomposition. When using the Tucker decomposition, only

$\mathcal{O}\left(R^D + DIR\right)$, $R = \max R_k$, $I = \max I_k$ elements should be stored instead of I^D components.

For higher-order $\underline{\mathbf{Y}} \in \mathbb{R}^{M \times J_1 \times \ldots \times J_D}$, unfolded tensor $\underline{\mathbf{Y}}$ along the first mode is used as $\mathbf{Y} \in \mathbb{R}^{M \times J}$ where $J = J_1 \cdot \ldots \cdot J_D$.

4 Numerical experiments

The proposed method of applying hankelization before dimensionality reduction was tested on real data. The goal of the experiment was to investigate whether hankelization helps to improve the quality of three different dimensionality reduction models for multiview BCI datasets. These models are MPCA, HOPLS and Hybrid Autoencoder (TensorReducedNet). In this chapter, datasets used in this work are described, metrics for performance evaluation are defined, and results are performed and discussed.

4.1 Datasets

In this section the datasets used in this work will be described. All datasets are multiview, as they have many channels. The datasets are differs in number of channels and method of receiving brain signals (EEG, ECoG).

4.1.1 Neurotycho dataset

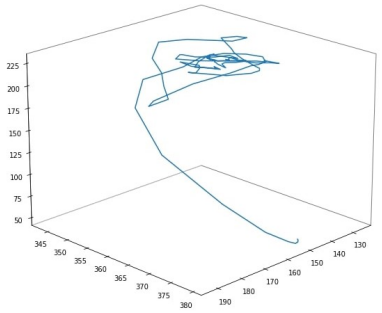


Figure 4.1. Hand movement trajectory

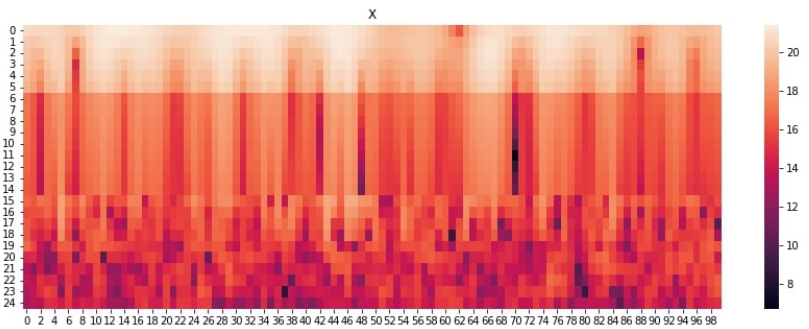


Figure 4.2. One channel data in time-frequency domain

One of the datasets for the computational experiment is electrocorticogram (ECoG) data from the Neurotycho [19] dataset. The data consist of 32-channel voltage signals taken from the monkey's brain. ECoG data are multidimensional and measurements correlate in both temporal and spatial domains. The target variable is the coordinates of the position of the hand in space (Fig. 4.1).

The original voltage signals were converted into a time-frequency representation using a wavelet transform with a parent Morlet wavelet, since this type of transformation is often used in problems with ECoG data [4], [2]. Frequencies were chosen as in [14]. The description of the source signal at each time had the dimension 32 (channels) $\times 27$ (frequencies) $= 864$. Each signal represented a local time interval with a duration of $\Delta t = 1s$. The time step between the signals is $\delta t = 0.05s$. The data had dimensions $\underline{\mathbf{X}} \in \mathbb{R}^{T \times 32 \times 27}$ (Fig. 4.2) and $\mathbf{Y} \in \mathbb{R}^{T \times 3}$. The data were divided into training and test samples in the ratio of $\frac{1}{3}$.

We applied hankelization to the initial data with $\tau_0 = 10$ by temporal dimension and with $\tau_1 = 2$ by spatial dimension. The shape of the hankelized tensors are in Table 4.1.

Also, we used data from Neurotycho [19] dataset with 64 channels, to investigate how hankelization influence quality of prediction for more complex data. The data with 64 channels were preprocessed the same way as data with 32 channels.

4.1.2 Multimodal EEG dataset

The dataset includes 60-channel electroencephalography (EEG), see Fig. 4.5-4.6, and 7-channel electromyography (EMG), see Fig. 4.3-4.4. Subjects performed different upper-extremity movement tasks. The EEG data were filtered by authors between 8 and 30 Hz (μ and β bands, respectively), known to be within the motor-related frequency range. Besides, artifact rejection was performed. The original EEG signals are converted into a space time representation using a wavelet transform with a Morlet wavelet. The frequencies were chosen between 8 and 30 Hz with step equals 1 Hz. So, $\underline{\mathbf{X}} \in \mathbb{R}^{T \times 64 \times 24}$, where 24 is a number of frequencies. $\mathbf{Y} \in \mathbb{R}^{T \times 6}$ is the EMG data. Hankelized tensors have the shape as showed in Table 4.1.

Table 4.1. Data shape for different types of hankelization.

Hankelization type	Neurotycho (ch=32)	Neurotycho (ch=64)	EEG
Without hankelization	$\underline{\mathbf{X}} \in \mathbb{R}^{T\times 32\times 27}$ $\mathbf{Y} \in \mathbb{R}^{T\times 3}$	$\underline{\mathbf{X}} \in \mathbb{R}^{T\times 64\times 27}$ $\mathbf{Y} \in \mathbb{R}^{T\times 3}$	$\underline{\mathbf{X}} \in \mathbb{R}^{T\times 60\times 24}$ $\mathbf{Y} \in \mathbb{R}^{T\times 6}$
Hankelization along space dimension	$\underline{\mathbf{X}} \in \mathbb{R}^{T\times 31\times 2\times 27}$ $\mathbf{Y} \in \mathbb{R}^{T\times 3}$	$\underline{\mathbf{X}} \in \mathbb{R}^{T\times 63\times 2\times 27}$ $\mathbf{Y} \in \mathbb{R}^{T\times 3}$	$\underline{\mathbf{X}} \in \mathbb{R}^{T\times 59\times 2\times 24}$ $\mathbf{Y} \in \mathbb{R}^{T\times 6}$
Hankelization along time dimension	$\underline{\mathbf{X}} \in \mathbb{R}^{T'\times 10\times 32\times 27}$ $\underline{\mathbf{Y}} \in \mathbb{R}^{T'\times 10\times 3}$	$\underline{\mathbf{X}} \in \mathbb{R}^{T'\times 10\times 64\times 27}$ $\underline{\mathbf{Y}} \in \mathbb{R}^{T'\times 10\times 3}$	$\underline{\mathbf{X}} \in \mathbb{R}^{T'\times 10\times 60\times 24}$ $\underline{\mathbf{Y}} \in \mathbb{R}^{T'\times 10\times 6}$
Hankelization along both dimensions	$\underline{\mathbf{X}} \in \mathbb{R}^{T'\times 10\times 31\times 2\times 27}$ $\underline{\mathbf{Y}} \in \mathbb{R}^{T'\times 10\times 3}$	$\underline{\mathbf{X}} \in \mathbb{R}^{T'\times 10\times 63\times 2\times 27}$ $\underline{\mathbf{Y}} \in \mathbb{R}^{T'\times 10\times 3}$	$\underline{\mathbf{X}} \in \mathbb{R}^{T'\times 10\times 59\times 2\times 24}$ $\underline{\mathbf{Y}} \in \mathbb{R}^{T'\times 10\times 6}$

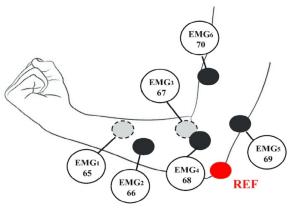


Figure 4.3. EMG electodes positions

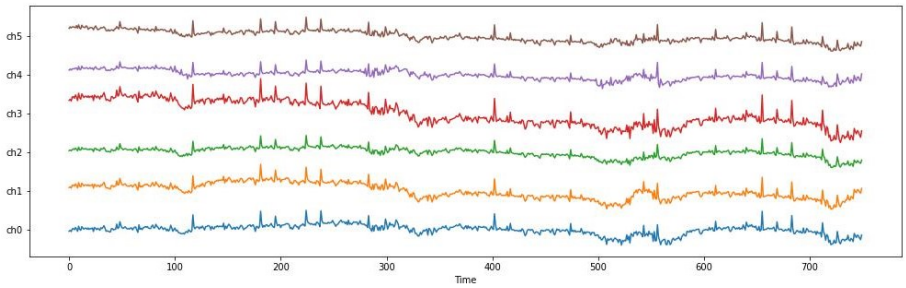


Figure 4.4. EMG data

4.2 Quality criteria

We used the scaled Root Mean Squared Error (sRMSE) to show the quality of the model prediction:

$$\text{sRMSE}(\underline{\mathbf{Y}}, \underline{\widehat{\mathbf{Y}}}_p) = \sqrt{\frac{\text{MSE}(\underline{\mathbf{Y}}, \underline{\widehat{\mathbf{Y}}}_p)}{\text{MSE}(\underline{\mathbf{Y}}, \overline{\underline{\mathbf{Y}}})}} = \frac{\|\underline{\mathbf{Y}} - \underline{\widehat{\mathbf{Y}}}_p\|_2}{\|\underline{\mathbf{Y}} - \overline{\underline{\mathbf{Y}}}\|_2}. \tag{4.2.1}$$

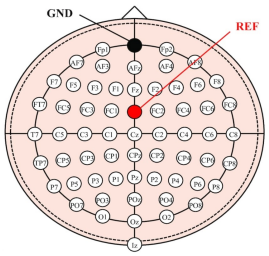


Figure 4.5. EMG electodes positions

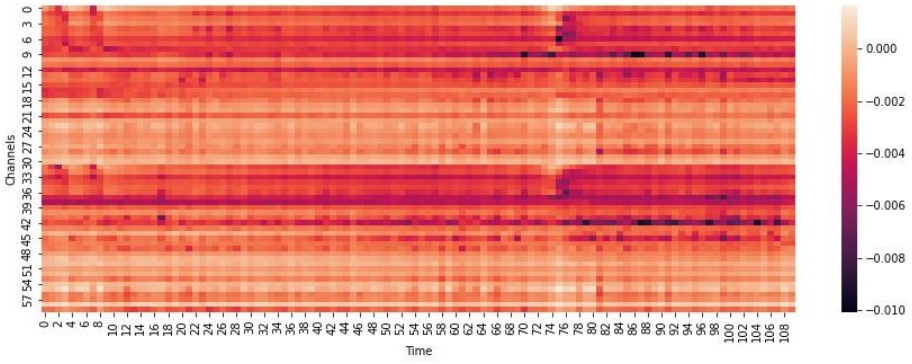


Figure 4.6. EEG data

Here $\underline{\widehat{\mathbf{Y}}}_p = \underline{\mathbf{X}}_p \underline{\mathbf{\Theta}}_p^{\mathsf{T}}$ is a model prediction and $\underline{\overline{\mathbf{Y}}}$ is a constant prediction obtained by averaging the targets across all objects. We tuned hyperparameters of the models by minimizing sRMSE on test sample.

To evaluate the complexity of models, we looked at shape of the latent representations of the initial data. Also, for graph to compare models, the number of values in latent representation of $\underline{\mathbf{X}}$, N_{values} was used. For latent tensor $\underline{\hat{\mathbf{X}}} \in \mathbb{R}^{L_1 \times \ldots \times L_G}$, N_{values} is calculated as follows:

$$N_{values} = L_1 \cdot \ldots \cdot L_G.$$

In order to simultaneously assess the complexity and accuracy of the model, we plot the graph in coordinates $(N_{values}, sRMSE)$. The model that is close to the left bottom corner of the graph is the optimal model.

In addition, we plotted examples of predicted trajectories with and without hankelization. It was done for better understanding of prediction quality.

4.3 Results

In this section, the results for three models and three datasets are presented. These results show that hankelization helps to improve the quality of the prediction and decreases the complexity of latent tensors in many cases. We divided each sample into ten subsamples of nine minutes length. We took two-thirds of each subsample as a train set and one third as a test set.

4.3.1 MPCA

We used MPCA as the most simple tensor dimensionality reduction model. We used MPCA that saves 97 of information. Shapes of latent variables for $\underline{\mathbf{X}}$ and $\underline{\mathbf{Y}}$ can be seen in the Table 4.2. Then, we trained Tucker tensor regression with ranks for each tensor mode as hyperparameters. Optimal hyperparameters were found by grid search. For this model, the optimal type of tensorization mostly is hankelization along space dimension. It can be seen in Fig. 4.8-4.9. However, no tensorization gives fewer number of values in latent representation of $\underline{\mathbf{X}}$. It can be noticed in Fig. 4.7-4.9.

Table 4.2. Results for different hankelization types obtained with the MPCA model.

Dataset	Hankelization	Shape of the latent variable of $\underline{\mathbf{X}}$	Shape of the latent variable of $\underline{\mathbf{Y}}$	sRMSE
Neurotycho (ch=32)	No	1×8	3	$\mathbf{1.727 \pm 0.043}$
	Along spatial dim	$10 \times 1 \times 8$	8×3	1.737 ± 0.042
	Along temporal dim	$1 \times 7 \times 2$	3	1.878 ± 0.067
	Along both dims	$10 \times 1 \times 7 \times 2$	8×3	1.912 ± 0.072
Neurotycho (ch=64)	No	1×27	3	4.241 ± 0.506
	Along spatial dim	$10 \times 1 \times 26$	4×3	4.067 ± 0.474
	Along temporal dim	$1 \times 22 \times 2$	3	4.173 ± 0.487
	Along both dims	$10 \times 1 \times 21 \times 2$	4×3	$\mathbf{4.038 \pm 0.439}$
EEG	No	1×10	3	1.766 ± 0.190
	Along spatial dim	$9 \times 1 \times 11$	9×4	$\mathbf{1.699 \pm 0.222}$
	Along temporal dim	$1 \times 14 \times 2$	3	2.499 ± 0.353
	Along both dims	$9 \times 1 \times 14 \times 2$	9×4	2.635 ± 0.498

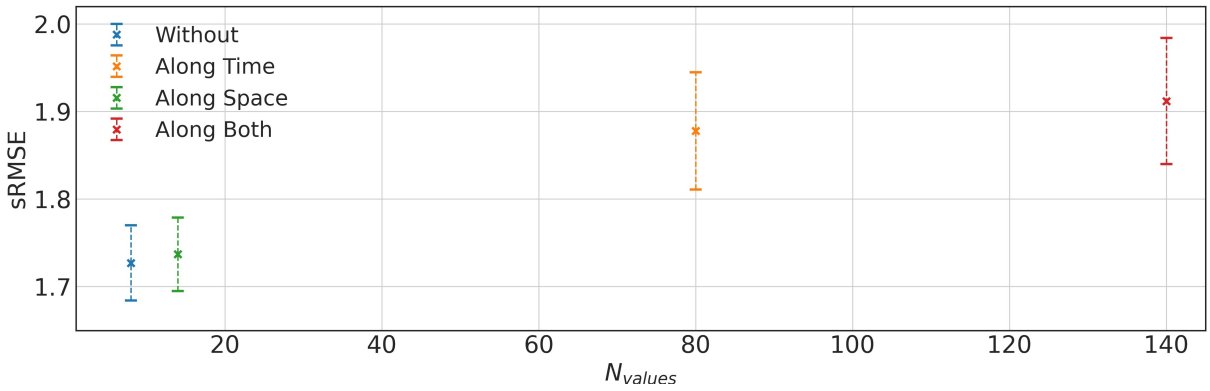


Figure 4.7. Comparison of hankelization types for Neurotycho dataset (ch=32) and the MPCA model.

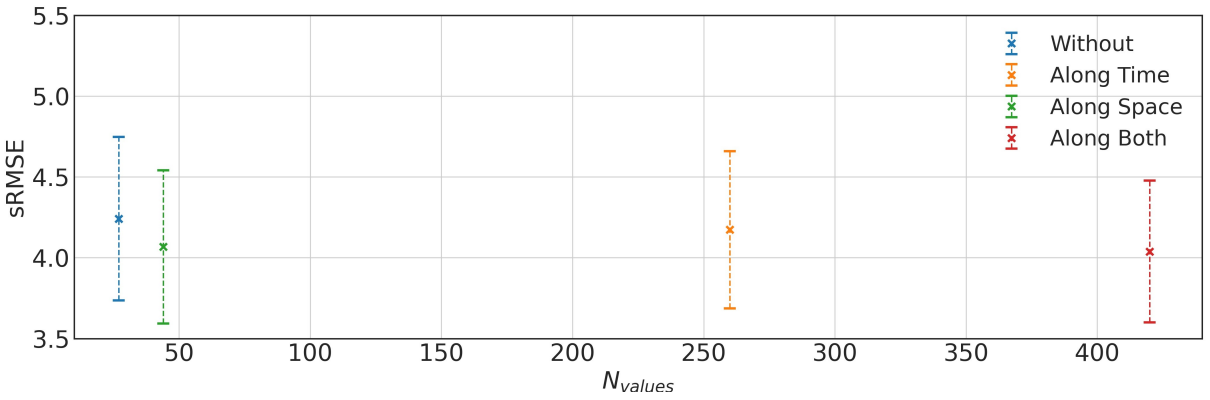


Figure 4.8. Comparison of hankelization types for Neurotycho dataset (ch=64) and the MPCA model.

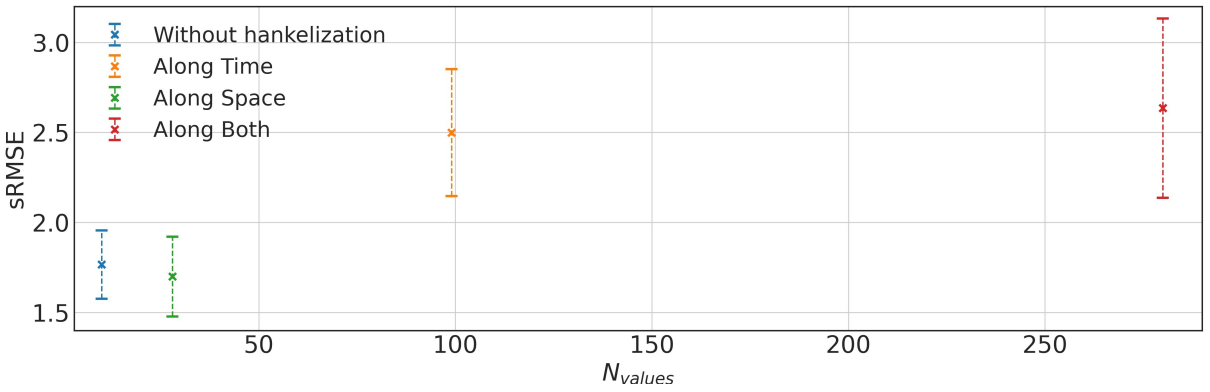


Figure 4.9. Comparison of hankelization types for Human EEG dataset (ch=60) and the MPCA model.

4.3.2 HOPLS

HOPLS is more complex model than MPCA as it makes an alignment between features and targets. We tuned the number of latent tensors by

maximizing the metric. We chose a metric connected with sRMSE 4.2.1 which equals $1 - sRMSE$. The hyperparameters of the HOPLS model are the shapes of latent representations of $\underline{\mathbf{X}}$ and $\underline{\mathbf{Y}}$. The optimal hyperparameters were found by grid search. The optimal shapes of core tensor of HOPLS are in Table 4.3. For two datasets, the optimal type of tensorization is hankelization along both dimensions. For the Human EEG dataset, the smallest metric was observed without hankelization. However, the fewest number of the values of latent representations was obtained by hankelization along space dimension. It can be seen in Fig. 4.10-4.12.

Table 4.3. Results for different hankelization types obtained with the HOPLS model.

Dataset	Hankelization	Shape of the latent variable of $\underline{\mathbf{X}}$	Shape of the latent variable of $\underline{\mathbf{Y}}$	sRMSE
Neurotycho (ch=32)	No	2×3	1	1.656 ± 0.050
	Along spatial dim	$\mathbf{3 \times 1 \times 1}$	1×1	1.444 ± 0.063
	Along temporal dim	$2 \times 2 \times 1$	1	$\mathbf{1.250 \pm 0.067}$
	Along both dim	$2 \times 3 \times 3 \times 1$	3×1	1.415 ± 0.054
Neurotycho (ch=64)	No	1×1	1	3.316 ± 0.424
	Along spatial dim	$2 \times 3 \times 1$	1×1	3.688 ± 0.451
	Along temporal dim	$\mathbf{2 \times 1 \times 2}$	1	3.044 ± 0.528
	Along both dims	$1 \times 2 \times 2 \times 2$	1×2	$\mathbf{2.782 \pm 0.499}$
EEG	No	1×8	1	1.324 ± 0.131
	Along spatial dim	$2 \times 3 \times 7$	2×1	$\mathbf{1.322 \pm 0.133}$
	Along temporal dim	$\mathbf{3 \times 1 \times 2}$	1	1.805 ± 0.196
	Along both dims	$2 \times 2 \times 5 \times 2$	1×4	1.797 ± 0.216

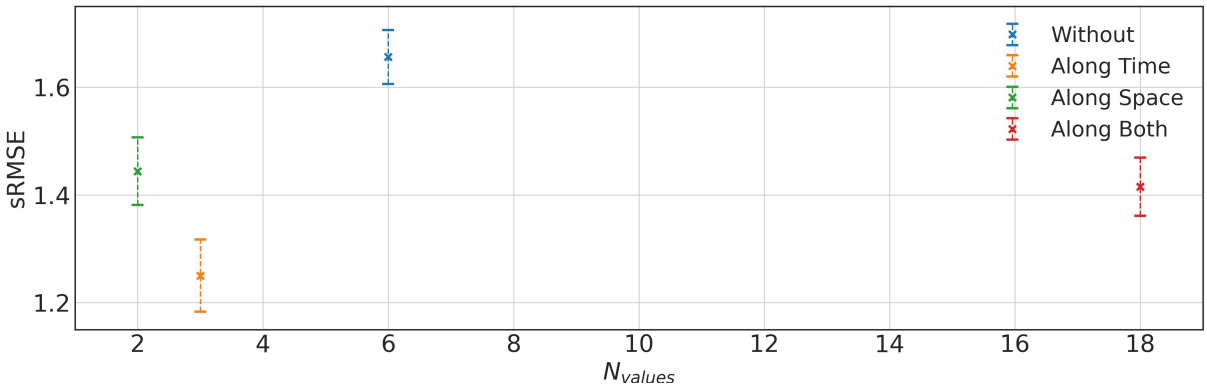


Figure 4.10. Comparison of hankelization types for Neurotycho dataset (ch=32) and the HOPLS model.

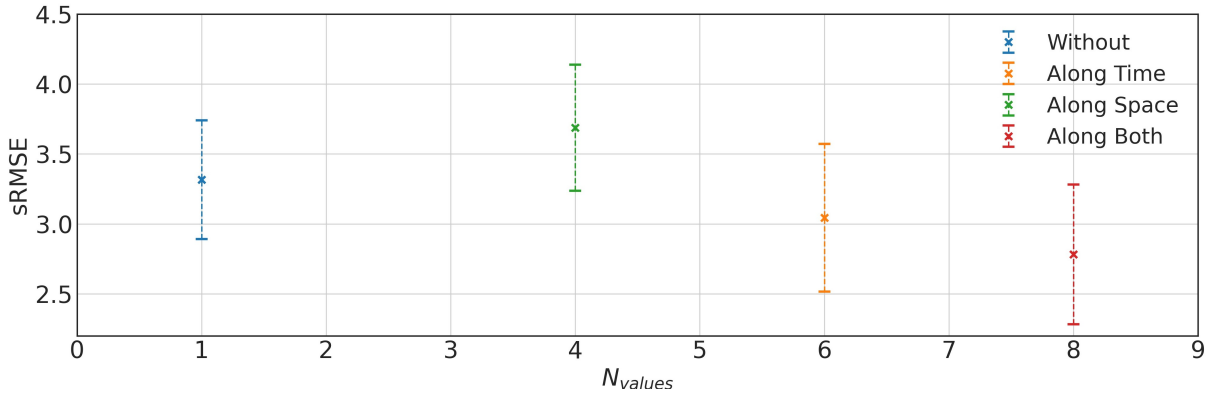


Figure 4.11. Comparison of hankelization types for Neurotycho dataset (ch=64) and the HOPLS model.

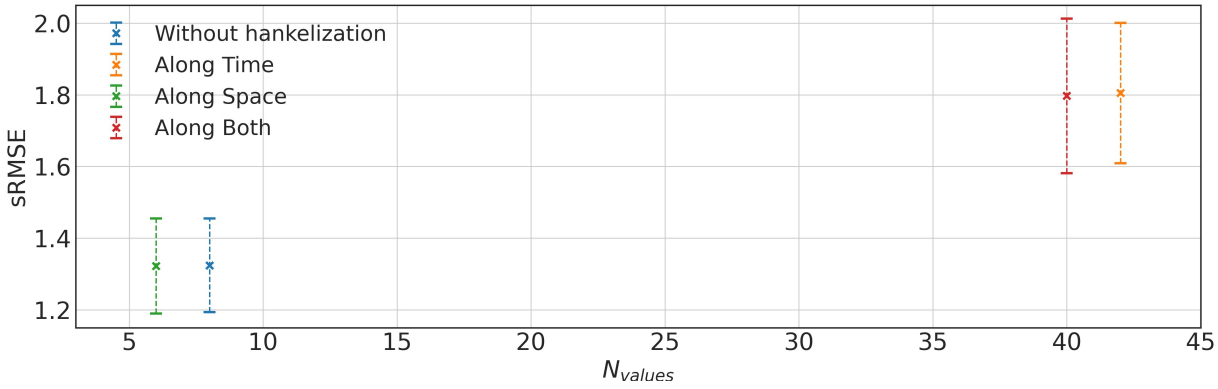


Figure 4.12. Comparison of hankelization types for Human EEG dataset (ch=60) and the HOPLS model.

4.3.3 TensorReducedNet

We decreased dimensionality only of $\underline{\mathbf{X}}$ by TensorReducedNet. This neural network was implemented with the Pytorch framework. The hyperparam-

eter of this model is the shape of the latent representation of $\underline{\mathbf{X}}$. It was tuned by grid search. For tuning parameters, we trained TensorReducedNet for 30 epochs and chose parameters that minimized sRMSE of predictions by tensor regression after dimensionality reduction by the encoder. Then TensorReducedNet was trained for more epochs.

The results are shown in Table 4.4. Hankelization along the space dimension helped to improve metric sRMSE for two out of three datasets. It also makes the smallest latent tensor that can be seen in Fig. 4.13-4.15.

Moreover, the coordinates of hand movement predicted by tensor regression after TensorReducedNet's encoder are in Fig. 4.16. These figures were obtained for one sample from the Neurotycho dataset with 64 channels. The figures demonstrate that hankelization along the spatial dimension gives similar results to results without hankelization. For this sample, the least noisy prediction seems to be the prediction obtained by hankelization along both dimensions.

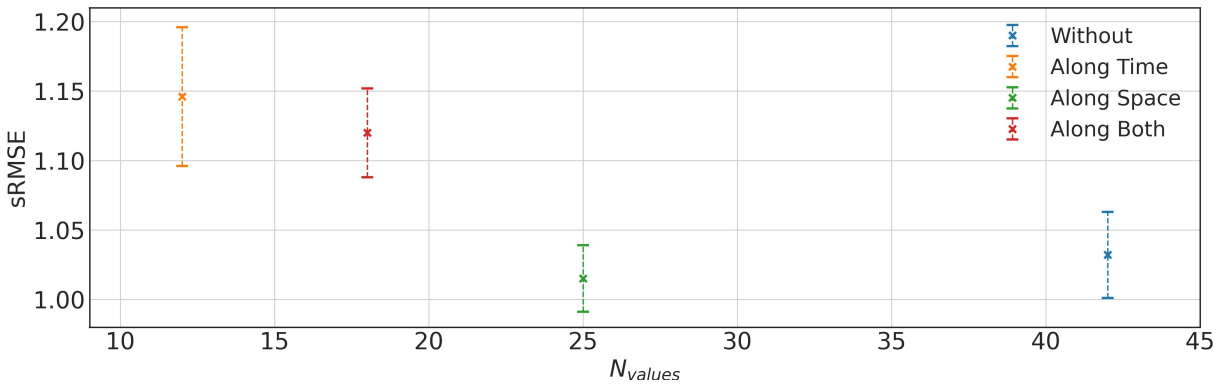


Figure 4.13. Comparison of hankelization types for Neurotycho dataset (ch=32) and the TensorReducedNet model.

Table 4.4. Results for different hankelization types obtained with TensorReducedNet.

Dataset	Hankelization	Shape of the latent variable of $\underline{\mathbf{X}}$	sRMSE
Neurotycho (ch=32)	No	7×6	1.032 ± 0.031
	Along spatial dimension	$5 \times 5 \times 1$	$\mathbf{1.015 \pm 0.024}$
	Along temporal dimension	$\mathbf{4 \times 3 \times 1}$	1.146 ± 0.050
	Along both dimensions	$3 \times 1 \times 3 \times 2$	1.120 ± 0.032
Neurotycho (ch=64)	No	8×10	$\mathbf{0.956 \pm 0.041}$
	Along spatial dimension	$\mathbf{8 \times 2 \times 1}$	1.003 ± 0.048
	Along temporal dimension	$3 \times 3 \times 2$	1.167 ± 0.084
	Along both dimensions	$3 \times 3 \times 3 \times 1$	1.117 ± 0.141
EEG	No	12×8	1.352 ± 0.151
	Along spatial dimension	$\mathbf{4 \times 1 \times 1}$	$\mathbf{1.338 \pm 0.105}$
	Along temporal dimension	$4 \times 1 \times 3$	1.636 ± 0.210
	Along both dimensions	$3 \times 1 \times 4 \times 2$	1.476 ± 0.260

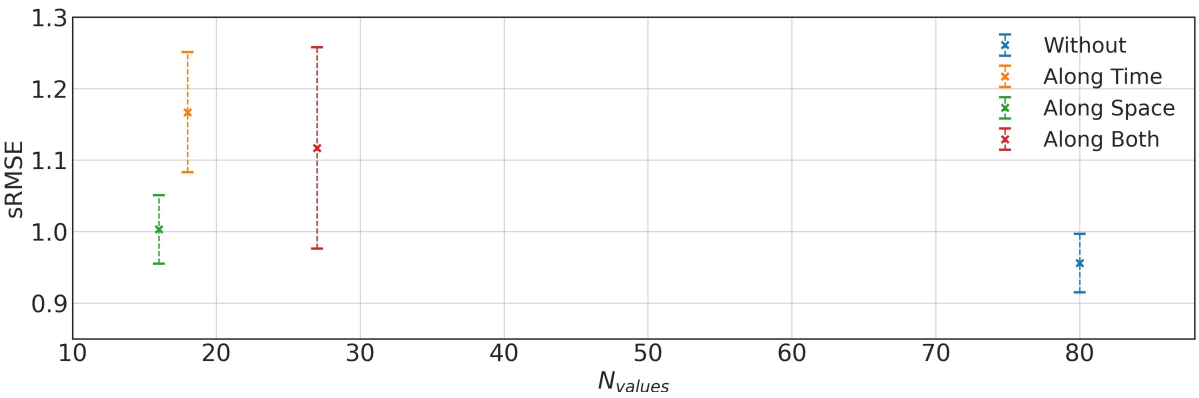


Figure 4.14. Comparison of hankelization types for Neurotycho dataset (ch=64) and the TensorReducedNet model.

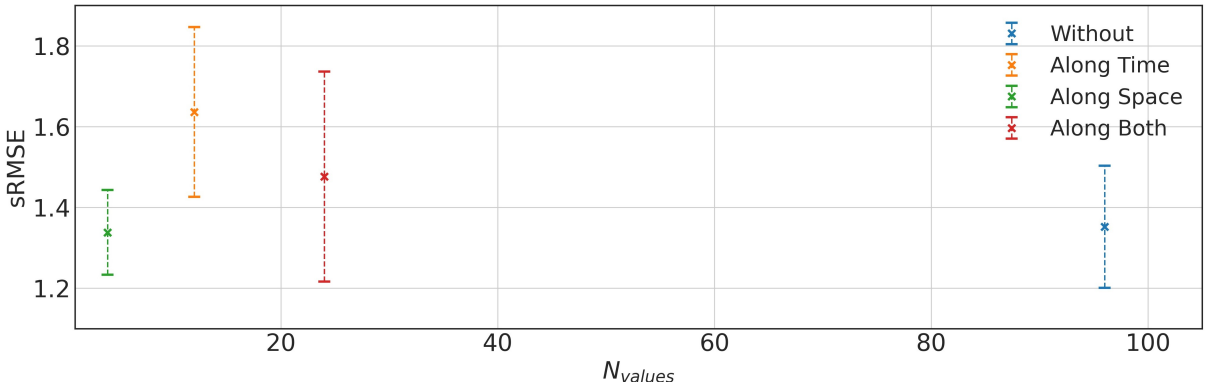
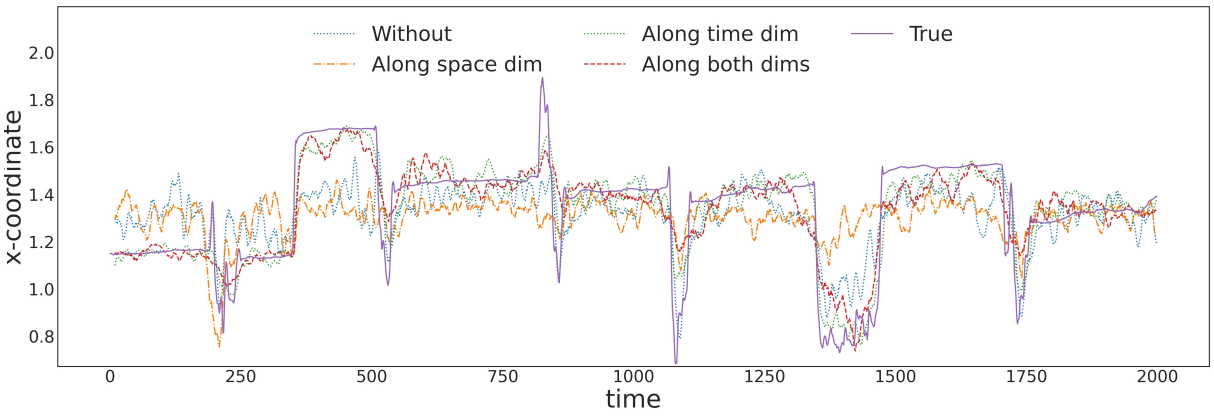
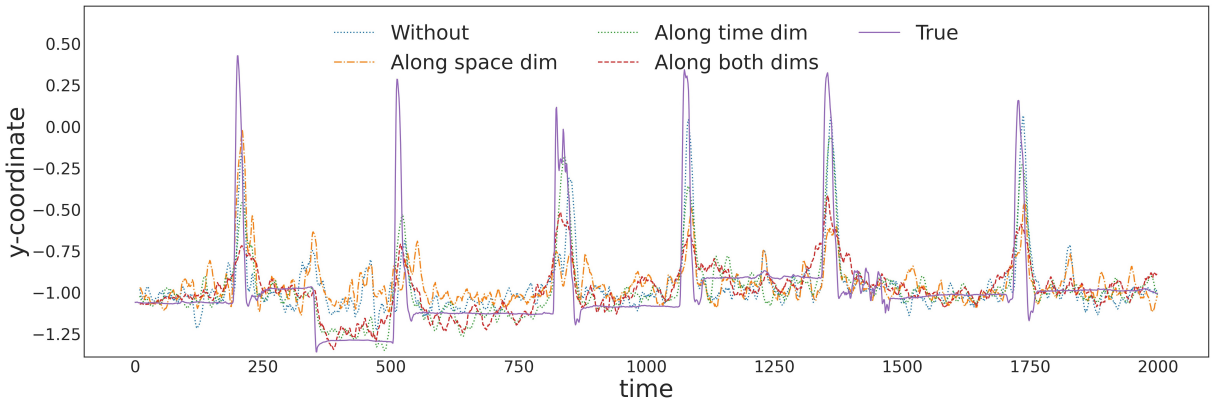


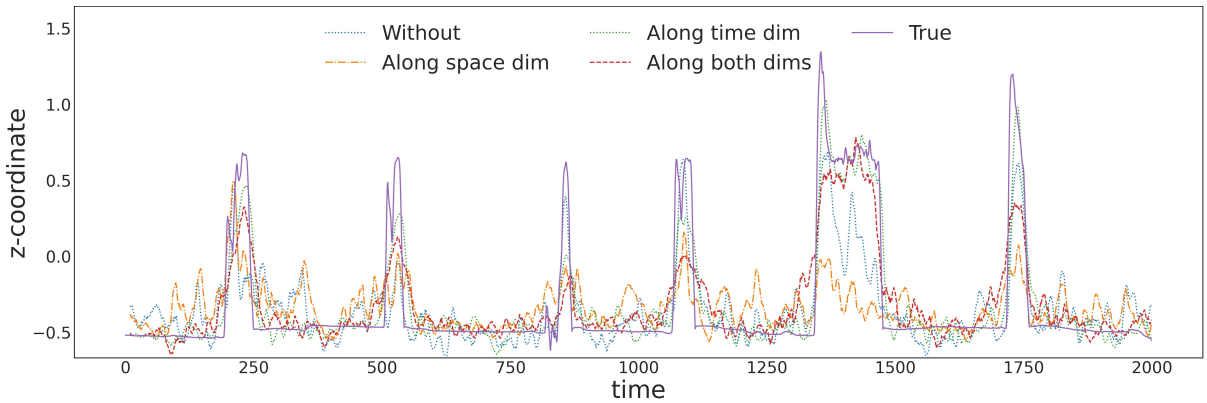
Figure 4.15. Comparison of hankelization types for Human EEG dataset (ch=60) and the TensorReducedNet model.



(а) x-axis



(б) y-axis



(в) z-axis

Figure 4.16. Example of 3D position of a hand predicted after dimensionality reduction made by Autoencoder with different types of hankelization.

5 Discussion and conclusion

In this work, the task of forecasting of target variables for Brain-Computer Interfaces was investigated. The data for BCI are usually correlated and redundant. To avoid instability of model because of these correlations, dimensionality reduction models are used. However, they usually do not take into account intrinsic correlations in time and space dimensions. To reveal this correlation, we propose to use hankelization before dimensionality reduction. We studied the influence of hankelization along temporal, along spatial and along both dimensions on forecasting quality. We tested hankelization on three datasets and before three dimensionality reduction models. These models are multilinear principal component analyses (MPCA), high-order partial least squares (HOPLS), and Autoencoder.

Previously, it was shown [18] that the forecasting of time series for other than BCI task is better with hankelization only along the temporal dimension. For BCI, our results showed that hankelization along the spatial dimension works the best way for MPCA and for AutoEncoder in many cases. For HOPLS, hankelization along temporal and spatial dimensions works better than only along the temporal dimension. It can be because of high correlation between data from different electrodes.

In future research, the optimal parameters τ_0 and τ_1 for hankelization along temporal and spatial dimensions should be found. In this work we worked with fixed τ_0 and τ_1 because of limited time. Moreover, future research should take into account the coordinates of electrodes before applying hankelization.

Bibliography

1. Y. Bengio and Yann Lecun. Convolutional networks for images, speech, and time-series. 11 1997.
2. Zenas C. Chao, Yasuo Nagasaka, and Naotaka Fujii. Long-term asynchronous decoding of arm motion using electrocorticographic signals in monkeys. Frontiers in Neuroengineering, 3, 2010.
3. Andrzej Cichocki, Anh-Huy Phan, Qibin Zhao, Namgil Lee, Ivan Oseledets, Masashi Sugiyama, Danilo P Mandic, et al. Tensor networks for dimensionality reduction and large-scale optimization: Part 2 applications and future perspectives. Foundations and Trends® in Machine Learning, 9(6):431–673, 2017.
4. A. Eliseyev and T. Aksenova. Penalized multi-way partial least squares for smooth trajectory decoding from electrocorticographic (ecog) recording. PloS one, page 11(5), 2016.
5. Simon Eyndhoven, Martijn Bousse, Borbala Hunyadi, Lieven Lathauwer, and Sabine Huffel. Single-channel EEG classification by multi-channel tensor subspacelearning and regression. IEEE interentional workshop on machine learning for signel processing, 2018.
6. Paul Geladi and Bruce R Kowalski. Partial least-squares regression: a tutorial. Analytica chimica acta, 185:1–17, 1986.
7. Roman Isachenko, Mariia Vladimirova, and Vadim Strijov. Dimensionality reduction for time series decoding and forecasting problems. DEStech Transactions on Computer Science and Engineering optim, pages 286–296, 2018.
8. Ji Hoon Jeong, Jeong Hyun Cho, Kyung Hwan Shim, Byoung Hee Kwon, Byeong Hoo Lee, Do Yeun Lee, Dae Hyeok Lee, and Seong Whan Lee. Multimodal signal dataset for 11 intuitive movement tasks from single upper extremity during multiple recording sessions. GigaScience, 9, 2020.
9. Tianyao Ji, Yuzi Jiang, Mengshi Li, and Qinghua Wu. Ultra-short--term wind speed and wind power forecast via selective hankelization and low-rank tensor learning-based predictor. International Journal of Electrical Power and Energy Systems, 140, 2022.
10. Ian T Jolliffe and Jorge Cadima. Principal component analysis: a

review and recent developments. Philosophical transactions of the royal society A: Mathematical, Physical and Engineering Sciences, 374(2065):20150202, 2016.

11. Steven Lemm, Benjamin Blankertz, Gabriel Curio, and Klaus Robert Müller. Spatio-spectral filters for improving the classification of single trial eeg. IEEE Transactions on Biomedical Engineering, 52, 2005.
12. Yingming Li, Ming Yang, and Zhongfei Zhang. A survey of multi-view representation learning. IEEE Transactions on Knowledge and Data Engineering, 31, 2019.
13. Haiping Lu, Konstantinos N Plataniotis, and Anastasios N Venetsanopoulos. Mpca: Multilinear principal component analysis of tensor objects. IEEE transactions on Neural Networks, 19(1):18–39, 2008.
14. A. Motrenko and V. Strijov. Multi-way feature selection for ecog-based brain-computer interface. Expert Systems with Applications, 114:402–413, 2018.
15. Akinari Onishi, Anh Huy Phan, Kiyotoshi Matsuoka, and Andrzej Cichocki. Tensor classification for p300-based brain computer interface. 2012.
16. X. Ran, W. Chen, B. Yvert, and S. Zhang. A hybrid autoencoder framework of dimensionality reduction for brain-computer interface decoding. Computers in Biology and Medicine, page 148, 2022.
17. Ziyu Shen, Binghui Liu, Qing Zhou, Zheng Liu, Bin Xia, and Yun Li. Cost-sensitive tensor-based dual-stage attention lstm with feature selection for data center server power forecasting. ACM Transactions on Intelligent Systems and Technology, 10 2022.
18. Qiquan Shi, Jiaming Yin, Jiajun Cai, Andrzej Cichocki, Tatsuya Yokota, Lei Chen, Mingxuan Yuan, and Jia Zeng. Block hankel tensor arima for multiple short time series forecasting. 2020.
19. K. Shimoda, Y. Nagasaka, Z. C. Chao, and N. Fujii. Decoding continuous three-dimensional hand trajectories from epidural electrocorticographic signals in japanese macaques. Journal of neural engineering, page 9(3), 2012.
20. Xudong Wang, Luis Miranda-Moreno, and Lijun Sun. Hankel-structured

tensor robust pca for multivariate traffic time series anomaly detection. 10 2021.

21. Hok Shing Wong, Li Wang, Raymond Chan, and Tieyong Zeng. Deep tensor cca for multi-view learning. IEEE Transactions on Big Data, 8, 2022.
22. Tatsuya Yokota, Burak Erem, Seyhmus Guler, Simon K. Warfield, and Hidekata Hontani. Missing slice recovery for tensors using a low-rank model in embedded space. 2018.
23. Qibin Zhao, Cesar F. Caiafa, Danilo P. Mandic, Zenas C. Chao, Yasuo Nagasaka, Naotaka Fujii, Liqing Zhang, and Andrzej Cichocki. Higher order partial least squares (hopls): A generalized multilinear regression method. IEEE Transactions on Pattern Analysis and Machine Intelligence, 35, 2013.
24. Qibin Zhao, Liqing Zhang, and Andrzej Cichocki. Multilinear and nonlinear generalizations of partial least squares: an overview of recent advances. Wiley Interdiscip. Rev. Data Min. Knowl. Discov., 4(2):104–115, 2014.
25. Guoxu Zhou, Qibin Zhao, Yu Zhang, Tulay Adali, Shengli Xie, and Andrzej Cichocki. Linked component analysis from matrices to high-order tensors: Applications to biomedical data. Proceedings of the IEEE, 104:310–331, 2 2016.