Single finger trajectory prediction from intracranial brain activity using Block-Term Tensor Regression with fast and automatic component extraction

Axel Faes*, Flavio Camarrone* and Marc M. Van Hulle, *Fellow, IEEE*

***Abstract*—Multiway- or tensor-based decoding techniques for Brain Computer Interfaces (BCI) are believed to better account for the multilinear structure of brain signals than conventional vector- or matrix-based ones. However, despite their outlook on significant performance gains, the used parameter optimization approach is often too computationally demanding so that conventional techniques are still preferred. We propose two novel tensor factorizations which we integrate into our Block-Term Tensor Regression (BTTR) algorithm and further introduce a marginalization procedure that guarantees robust predictions while reducing the risk of overfitting (generalized regression). BTTR accounts for the underlying (hidden) data structure in a fully automatic and computationally-efficient manner, leading to a significant performance gain over conventional vector- or matrix-based techniques in a challenging real-world application. As a challenging real-world application, we apply BTTR to accurately predict single finger movement trajectories from intracranial recordings in human subjects. We compare the obtained performance with that of the state-of-the-art.**

***Index Terms*—tensor decomposition, partial least squares, multiway data, tensor, block-term decomposition, brain computer interfaces, electrocorticography, finger movement decoding**

I. Introduction

BRAIN Computer Interfaces (BCIs) decode the user's intent from his/her brain activity directly, bypassing the need for muscular control. Detection requires computational processing to separate ("decode") activity related to the intended action from background noise after which the result is sent to an actuator (see review [1]). The ability to control a robotic limb or regain control over a paralyzed limb with "motor-BCIs" has promoted the latter as a solution for patients deprived from voluntary movement, but that are otherwise fully conscious, due to spinal cord injury, brain stem stroke or a degenerative disorder such as amyotrophic lateral sclerosis (ALS). Extraordinary results have been achieved with functional electrical stimulation (FES) of hand muscles [2], [3] and prosthetic hands and arms, exoskeletons or other effectors [4], [5], [6], [7].

In the last decade, BCI researchers are paying increased attention to electrocorticography (ECoG) where brain activity is recorded via grids or strips of electrodes placed intracranially on the cortical surface. Compared to multi-electrode and deep brain implants, ECoG offers larger coverages of cortical surface combined with more long-term signal stability [8], [9] and, compared to electroencephalography (EEG), a higher spatial resolution [10] and a larger spectral bandwidth and signal amplitude [11]. Several studies [12], [13], [14], [15], [16], [17], [18] reported that ECoG signals recorded from the primary motor cortex can promote successful decoding of continuous arm trajectories. Nevertheless, accurate decoding of more subtle muscular activity such as single finger extension/flexion is considered the next challenge.

Currently, ECoG-based decoding approaches for finger movements have been proposed based mainly on conventional sparse [19], gaussian [20], linear regression [21], [22], Convolutional Neural Networks (CNN), Random Forests (RF) and deep learning networks (proposed in [23]). However, despite the encouraging results, we advocate that finger trajectory decoding could benefit from recent developments in multilinear algebra (multiway decoding) as ECoG is in essence structured in space, time, and frequency domain. For instance, in relation to muscular actions, a distinctive spatio-temporal-spectral activity pattern can be observed in the motor cortex: a decrease in activity in both the mu and beta bands following movement onset termed event-related desynchronisation (ERD) (Graimann et al., 2011), and an increase in beta band activity following movement termination termed event-related synchronisation (ERS) [24], whereby the spatial distribution of both depends on the considered movement task. When relying on a conventional vector- or matrix-based regression model, the original multimodal structure is largely ignored as the data is concatenated into vectors and matrices (aka unfolding), which could cause the model to underperform and hamper its interpretation [25]. Furthermore, the size of the vectorized/matricized data calls for a model with a large number of tunable parameters rendering it susceptible to noise and redundancy [26]. Modern deep learning methods have also been proposed, such as those based on Riemannian features [27].

On the other hand, multiway models preserve the multilinear structure of the data and support the discovery of potentially hidden multilinear components [28]. For this reason, they have attracted interest from researchers in neuroimaging, image and video completion, numerical analysis, data mining, etc. The two most popular multiway frameworks are Tucker (TKD) and CANDECOMP/PARAFAC (or CPD) decompositions (see

Axel Faes (corresponding author) was with the Laboratory for Neuro- and Psychophysiology, Department of Neurosciences, KU Leuven, Leuven, Belgium.
E-mail: axel.faes@kuleuven.be

Flavio Camarrone was with the Laboratory for Neuro- and Psychophysiology, Department of Neurosciences, KU Leuven, Leuven, Belgium.

Marc M. Van Hulle was with the Laboratory for Neuro- and Psychophysiology, Department of Neurosciences, KU Leuven, Leuven, Belgium.

* Equal contribution

[28], [29] for review) and both aim to determine the low-dimensional space where important information is residing. While the former seeks a low multilinear rank (MTR) approximation of the tensor data, the latter approximates the tensor data as a sum of rank-one tensors. Note that the concept of rank is associated with the "information content" of the tensor: the lower the rank, the lower the information content. Clearly, identifying the true rank of a tensor is the key to accurate multiway analysis. De Lathauwer introduced a tensor decomposition algorithm, called Block Term Decomposition (BTD) [30], [31], [32], which can be seen as a unified version of TKD and CPD. More specifically, in [32] two BTD models were proposed to approximate N^{th}-order tensor data as a sum of K-blocks called rank-$(L_1^k, ..., L_N^k)$ BTD and rank-$(L_1, ..., L_N)$ BTD with k=1,...,K. In the former model blocks can have different MTR, whence in the latter a unique MTR is chosen for all blocks. If there is only one block, then BTD reduces to TKD. In contrast, if all blocks have multilinear rank MTR = $(1, 1, ..., 1)$, then it reduces to CPD. This approach provides great flexibility and opens new possibilities for multiway data analysis, above all when rank-$(L_1^k, ..., L_N^k)$ BTD is adopted. For instance, a recent EEG study [33] showed that, while CPD failed to model epileptic seizures, rank-$(L_1, ..., L_N)$ BTD correctly extracted the ictal sources that matched clinical assessment.

The abovementioned multiway frameworks have been adapted for regression analysis to model the relationship between arm trajectory and ECoG signals recorded from monkeys [34], [35] and recently between exoskeleton-based arm trajectory, arm- and wrist rotations and ECoG signals recorded from a tetraplegic patient with a spinal cord injury [18]. The decoder in the latter case relies on a variant of multiway partial least squares regression (NPLS), but is not capable of achieving the kind of accuracy needed to decode fine limb movements. Possible reasons for this underperformance are the limited fitness ability, the high computational complexity and the slow convergence of NPLS when handling higher-order data [34]. On the other hand, Zhao et al. [34] developed a powerful generalized framework, called Higher-Order Partial Least Squares (HOPLS), based on $(1, L_2, ..., L_N)$-rank BTD (i.e. all blocks have the same MTR), that provides enhanced predictability with optimal balance between fitness and model complexity. As a result, HOPLS was able to outperform conventional PLS.

However, the success of HOPLS and other tensor-based regression methods largely depends on the selection of appropriate model parameters, e.g., the number of latent variables, a challenging issue and the subject of ongoing research. Evidently, cross-validation could be used to identify the (near) optimal parameter combination by assessing all (or as many as desired) parameter combinations, but the ensuing computational effort could be too prohibitive to be practical, above all in time-critical applications such as BCI. In addition, as current multiway regression models use fixed MTRs, they do not fully exploit the potential of rank-$(L_1^k, ..., L_N^k)$ BTD, above all in cases where the true rank changes across blocks.

In order to accurately decode single finger movements from ECoG signals, a challenging real-world application, we propose a fast and flexible multiway model based on $(L_1^k, ..., L_N^k)$ BTD (namely BTTR) with automatic parameter selection. In this way, we tackle the computationally intensive parameter estimation assumed by state-of-the-art tensor-based methods as well as cases where the optimal MTR varies across blocks. Further we introduce a marginalization procedure to obtain robust predictions while reducing the risk of overfitting (generalized regression). Our motivation for both developments is to promote tensor-based techniques in particular for BCI purposes. In the next sections, we introduce a new TKD method for automatic rank selection and latent component extraction given two more or less correlated multiway variables, called Automatic Component Extraction (ACE), and its improved version augmented with automatic latent component selection, called Automatic Correlated Component Selection (ACCoS). The ACE or ACCoS models are embedded in the BTTR process to sequentially extract the components that maximize the correlation between multiway response and prediction. We then applied BTTR on ECoG data taken from BCI competition IV [36]. As a result, our model yields a higher accuracy compared to HOPLS, linear regression [22], Random Forests (RF), Convolutional Neural Networks (CNN), and Long Short-Term Memory Network (LSTM) [23]. In this way, we wish to promote the use of multiway modelling for real-world BCI applications where accurate and fast decoding of brain activity is required as in ECoG-based neuroprostheses.

II. Previous work on automatic multilinear tensor rank selection

Large tensor data usually contains intrinsically low- dimensional information extractable with tensor frameworks such as TKD, a form of higher-order principal component analysis that is adopted in a broad range of applications [37]. Given a tensor $\underline{\mathbf{X}} \in \mathbb{R}^{I_1 \times ... \times I_N}$, TKD decomposes $\underline{\mathbf{X}}$ into a smaller tensor (core tensor) and N factor matrices: $\underline{\mathbf{X}} \approx \underline{\mathbf{G}} \times_1 \mathbf{A}^{(1)} \times_2 ... \times_N \mathbf{A}^{(n)}$ with $\underline{\mathbf{G}} \in \mathbb{R}^{R_1 \times R_n}$ the core tensor and $\mathbf{A}^{(n)} \in \mathbb{R}^{I_n \times R_n}$ the factor matrix associated with the nth mode. The obtained decomposition is often not unique so that constraints such as orthogonality, sparsity or nonnegativity are generally imposed on the factor matrices and/or core tensor [38], [39], [40]. For instance, Higher-Order Singular Value Decomposition (HOSVD) [38] and Higher-Order Orthogonal Iteration (HOOI) [41] were proposed as variants of TKD with orthogonality constraints on the factor matrix.

The multilinear rank (MTR) is the tuple $R_1 \times ... \times R_n$ that defines the size of the factor matrices and, hence, the core tensor. The choice of MTR is generally considered critical: a large MTR might lead to an approximation containing uninteresting information; a small MTR might yield a low compression ratio incapable of fully representing complex tensor data. Recently, researchers have been focusing on determining the MTR, and thus model complexity, in an automatic way. For instance, Mørup and Hansen [42] proposed a method based on Bayesian learning for sparse Tucker decomposition. In this method, called ARD Tucker, the core tensor and matrices are alternately and iteratively updated, while the number of components in each mode are determined using automatic relevance determination (ARD). Yokota et al. [43] proposed pruning sparse Tucker decomposition (PSTD) of which the objective is to minimize the L1-norm of the core tensor under conditions of error bound and orthogonality constraints of individual basis matrices. At each iteration, the factor matrices and the sparsity of the core tensor are updated, and unnecessary dimensions removed according to the entries of the latter. More recently, Yokota et al [44] published a Tucker-rank estimation approach using robust minimum description length (MDL): the rank R_n is estimated by applying the MDL criterion on the distribution of the eigenvalues extracted from the nth mode unfolded core tensor via HOSVD. This method is referred to as SCORE. Shi et al. [45] proposed a multilinear Tensor Rank Estimation based on L1-regularized orthogonal CP decomposition (TREL1). Using a block coordinate descent approach, the CP components and corresponding weight vectors are iteratively updated. Finally, TREL1 automatically determines the MTR by pruning the zero entries of the weight vector.

III. Methods

One of the main limitations of HOPLS, the current state-of- the-art multiway regression models, is the assumed prior knowledge of the model parameters, but this information is commonly not available for a given real-word application. Therefore, the computationally expensive cross-validation is often used to identify these parameters. In addition, HOPLS can be seen as a BTD-like approach where each block has the same $(1, L_2, ..., L_N)$-rank which, in turn, limits flexibility in data modelling. To tackle this, we first define a novel tensor decomposition model with automatic MTR determination with the intent to extract the components that are maximally correlated between two variables. Then, we include the proposed model into a deflation-based method for multiway regression, namely BTTR (Block Term Tensor Regression), for high flexibility and interpretability. An overview of the mathematical notation used can be found in Table I.

Using BTTR first requires formatting the data into a tensor format. BTTR will work iteratively. Each iteration either ACE or ACCoS is used for automatic parameter estimation. This is followed by deflation before the next iteration starts.

A. Component extraction and selection

Given an N-way variable $\underline{\mathbf{X}} \in \mathbb{R}^{I_1 \times ... \times I_N}$ and a vectoral variable $\mathbf{y} \in \mathbb{R}^{I_1 \times 1}$, we aim to automatically extract the latent components $\mathbf{t}$ and $P^(n)_{(n=2)}^N$, associated with the n-th mode of $\underline{\mathbf{X}}$ and maximally correlated with $\mathbf{y}$, while $\underline{\|\underline{\mathbf{X}}\| - [\![\underline{\mathbf{G}}; \mathbf{t}, \mathbf{P}^{(2)}, ..., \mathbf{P}^{(N)}]\!]}_F$ is minimized. To this end, we first introduce a novel model for tensor decomposition, characterized by robust component extraction. Then, we extend the proposed model with correlated component selection. An overview of the proposed models is reported in Figure 1.

1) Modified PSTD for component extraction

Our new method for determining MTR uses sparsity constrains on the core tensor to prune irrelevant components. This idea is inherited from the PTSD model which has recently proved to be a valid tool for MTR selection. We will refer to the proposed model as modified PSTD (mPSTD). The mPSTD model is first initialized with HOSVD. Then, iteratively, a soft-thresholding rule based on parameter λ, alternated with a threshold τ –adopted from standard PSTD–, are applied to enhance model sparsity and to prune irrelevant components, respectively. Note that in [43] SNR $\in [1, 50]$ is used to derive, via a line search, the optimal degree of sparsity λ of the core tensor (see [43] for a derivation of λ). At each iteration, the core tensor $\underline{\mathbf{G}}$ is updated using the soft-thresholding rule as $\underline{\mathbf{G}} = sgn(\underline{\mathbf{G}}) \times max\{|\underline{\mathbf{G}}| - \lambda, 0\}$, while the threshold $\tau \in [0, 100]$ is used to reject unnecessary components from the n-mode $S^{(n)} = \{r|100(1 - \frac{\sum_i \mathbf{G}_{(n)(r,i)}}{\sum_{t,i} \mathbf{G}_{(n)(t,i)}}) \geq \tau\}$, $\mathbf{P}^{(n)} = \mathbf{P}^{(n)}(:, S^{(n)})$ and $\mathbf{G}^{(n)} = \mathbf{G}^{(n)}(S^{(n)}, :)$. The mPTSD is summarized in Algorithm 1.

2) Proposed ACE for automatic component extraction

Importantly, in standard and modified PSTD, parameters such as noise level of the data (SNR) and the threshold for component rejection (τ) are assumed to be known. However, in reality, this is not always the case and can have an important impact on model performance. Let us now define the mode-1 cross-product between predictor and response variables as $\underline{\mathbf{C}} = \langle \underline{\mathbf{X}}, \mathbf{y} \rangle_{(1)} \in \mathbb{R}^{1 \times I_2 \times ... \times I_N}$ and its decomposition as

TABLE I
Mathematical notation

Notation	Description
$\underline{\mathbf{T}}, \mathbf{M}, \mathbf{v}, S$	tensor, matrix, vector, scalar (respectively)
$\mathbf{M}^T$	transpose of matrix
$\times_n$	mode-n product between tensor and matrix
$\otimes$	Kronecker product
$\circ$	outer product
$\|\cdot\|_F$	Frobenius norm
$\mathbf{T}_{(n)}$	mode-n unfolding of tensor $\underline{\mathbf{T}}$
$\underline{\mathbf{C}}^{(T)}$	core tensor associated to tensor $\underline{\mathbf{T}}$
$\mathbf{M}^{(n)}$	mode-n factor matrix
$\mathbf{M}_{ind}$	(sub-)matrix including the column(s) indicated in ind
$\mathbf{M}_{\setminus ind}$	(sub-)matrix excluding the column(s) indicated in ind
$[\![\underline{\mathbf{C}}; \mathbf{M}^{(1)}, ..., \mathbf{M}^{(N)}]\!]$	full multilinear product $\underline{\mathbf{C}} \times_1 \mathbf{M}^{(1)} \times_2 \cdots \times_N \mathbf{M}^{(N)}$
$\langle \underline{\mathbf{T}}, \underline{\mathbf{E}} \rangle_{\{n,n\}}$	mode-n cross-covariance tensor

```
Input: $\underline{\mathbf{X}} \in \mathbb{R}^{I_1 \times ... \times I_N}, \mathbf{y} \in \mathbb{R}^{I_1 \times 1}, \tau$, SNR
Output: $\underline{\mathbf{G}} \in \mathbb{R}^{1 \times R_2 \times ... \times R_N}, \{\mathbf{P}^{(n)}\}_{n=2}^N$
    Initialisation :
1:  $\underline{\mathbf{C}} = \langle \underline{\mathbf{X}}, \mathbf{y} \rangle_{(1)} \in \mathbb{R}^{1 \times I_2 \times ... \times I_N}$
2:  Initialisation of $\{\mathbf{P}^{(n)}\}_{n=2}^N$ and $\underline{\mathbf{G}}$ using HOOI on $\underline{\mathbf{C}}$
    LOOP Process
3:  repeat
4:      update $\underline{\mathbf{G}}$ using SNR
5:      prune $\{\mathbf{P}^{(n)}\}_{n=2}^N$ and $\underline{\mathbf{G}}$ using $\tau$
6:  until convergence is reached
7:  return $\underline{\mathbf{G}}, \{\mathbf{P}^{(n)}\}_{n=2}^N$
```

Fig. 1. mPSTD

```
Input: $\underline{\mathbf{X}} \in \mathbb{R}^{I_1 \times \ldots \times I_N}, \mathbf{y} \in \mathbb{R}^{I_1 \times 1}$
Output: $\underline{\mathbf{G}}^{(X)} \in \mathbb{R}^{1 \times R_2 \times \ldots \times R_N}, \mathbf{t}, \{\mathbf{P}^{(n)}\}_{n=2}^{N}$
 1: $\underline{\mathbf{C}} = \langle \underline{\mathbf{X}}, \mathbf{y} \rangle_{(1)} \in \mathbb{R}^{1 \times I_2 \times \ldots \times I_N}$
 2: Initialisation of $\tau = 90, \ldots, 100$; $SNR = 1, \ldots, 50$
 3: for $SNR_i$ in $SNR$ do
 4:     for $\tau_j$ in $\tau$ do
 5:         $\underline{\mathbf{G}}, \{\mathbf{P}^{(n)}\}_{n=2}^{N}$ = $mPSTD(\underline{\mathbf{X}}, \mathbf{y}, SNR_i, \tau_j)$
 6:         calculate BIC value corresponding to $SNR_i$ and $\tau_j$
            using Eq 1
 7:     end for
 8:     select $\tau^*$ = $\mathrm{argmin}_\tau \mathrm{BIC}(\tau)$
 9:     calculate BIC value corresponding to $SNR_i$ and $\tau^*$
        using Eq 1
10: end for
11: select $SNR^*$ = $\mathrm{argmin}_{SNR} \mathrm{BIC}(SNR, \tau^*)$
12: $\underline{\mathbf{G}}, \{\mathbf{P}^{(n)}\}_{n=2}^{N}$ = $mPSTD(\underline{\mathbf{X}}, \mathbf{y}, SNR^*, \tau^*)$
13: $\mathbf{t} = (\underline{\mathbf{X}} \times_2 \mathbf{P}^{(2)T} \times_3 \ldots \times_N \mathbf{P}^{(N)T})_{(1)} vec(\underline{\mathbf{G}})$
14: $\mathbf{t} = \mathbf{t}/\|\mathbf{t}\|_F$
15: $\underline{\mathbf{G}}^{(X)} = [\![ \underline{\mathbf{X}}; \mathbf{t}^T, \mathbf{P}^{(2)T}, \ldots, \mathbf{P}^{(N)T} ]\!]$
16: return $\underline{\mathbf{G}}^{(X)}, \mathbf{t}, \{\mathbf{P}^{(n)}\}_{n=2}^{N}$
```

Fig. 2. ACE

$\underline{\mathbf{C}} \approx [\![\underline{\mathbf{G}}^{(c)}; \mathbf{P}^{(2)}, \ldots, \mathbf{P}^{(N)}]\!]$. We provide the model with automatic SNR and τ selection based on Bayesian Information Criterion (BIC) defined here as:

$$BIC(\tau, \mathrm{SNR}|\mathrm{SNR}, \tau^*) = \log\left(\frac{\|\underline{\mathbf{C}}\| - [\![\underline{\mathbf{G}}^{(c)}; \mathbf{P}^{(2)}, \ldots, \mathbf{P}^{(N)}]\!]_F}{s}\right) + \frac{\log(s)}{s} DF \quad (1)$$

where $\underline{\mathbf{G}}^{(c)}$ and $\{\mathbf{P}^{(n)}\}_{n=2}^{N}$ are the sparse core and factor matrices obtained with mPSTD using specific τ and SNR values, s the number of entries in $\underline{\mathbf{G}}$, and DF the degree of freedom calculated as the number of non-zero elements in $\underline{\mathbf{G}}^{(c)}$, as suggested in [46]. BIC is well known for its consistency in selecting the true model [47] as it is based on a trade-off between model fit and -complexity. A lower BIC value indicates a better candidate. For each SNR value, the associated optimal τ is computed as $\tau^* = \mathrm{argmin}_\tau BIC(\tau, \mathrm{SNR})$. Then, the optimal SNR is determined as $\mathrm{SNR}^* = \mathrm{argmin}_{\mathrm{SNR}} BIC(\mathrm{SNR}, \tau^*)$. Once $\underline{\mathbf{G}}^{(c)}$ and $\{\mathbf{P}^{(n)}\}_{n=2}^{N}$ are computed, the score vector t is first calculated as

$$\mathbf{t} = (\underline{\mathbf{C}} \times_2 \mathbf{P}^{(2)T} \times_3 \ldots \times_N \mathbf{P}^{(n)T})_{(1)} \mathrm{vec}(\underline{\mathbf{G}}^{(c)}),$$

and then normalized. We refer to this fully automatic component extraction as ACE, summarized in Algorithm 2.

3) Automatic correlated component selection (ACCoS)

Once the core tensor $\underline{\mathbf{G}}^{(c)}$ and factors $\{\mathbf{P}^{(n)}\}_{n=2}^{N}$ are extracted via ACE, we select only the relevant components in a fully automatic manner as well. The full process consists of two steps: scoring (Step 1) and grouping (Step 2). In Step 1, for each n-mode factor, the single rth component is scored using the R-squared test between $\mathbf{x}_r^{(n)}$ and $\mathbf{y}$ where

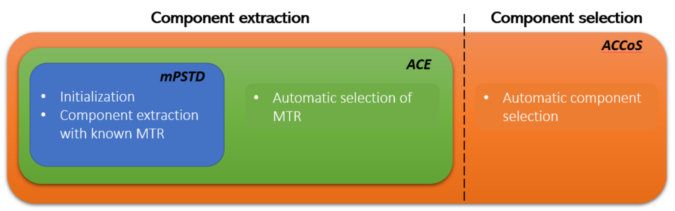

Component extraction
Component selection
ACCoS
ACE
mPSTD
Initialization
Component extraction with known MTR
Automatic selection of MTR
Automatic component selection

Fig. 3. Overview of the proposed models for component extraction and selection.

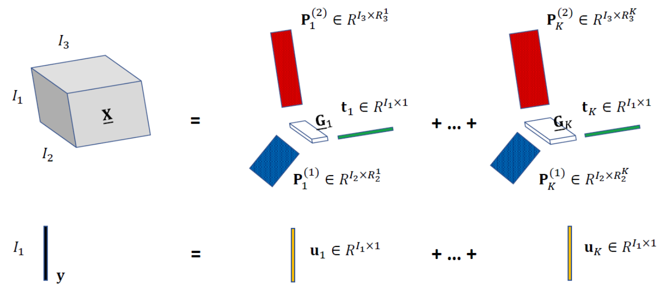

I_3
I_1
I_2
$\underline{\mathbf{X}}$
=
$\mathbf{P}_1^{(2)} \in R^{I_3 \times R_3^1}$
$\mathbf{t}_1 \in R^{I_1 \times 1}$
$\underline{\mathbf{G}}_1$
$\mathbf{P}_1^{(1)} \in R^{I_2 \times R_2^1}$
+ ... +
$\mathbf{P}_K^{(2)} \in R^{I_3 \times R_3^K}$
$\mathbf{t}_K \in R^{I_1 \times 1}$
$\underline{\mathbf{G}}_K$
$\mathbf{P}_K^{(1)} \in R^{I_2 \times R_2^K}$
I_1
$\mathbf{y}$
=
$\mathbf{u}_1 \in R^{I_1 \times 1}$
+ ... +
$\mathbf{u}_K \in R^{I_1 \times 1}$

Fig. 4. Scheme of BTTR algorithm with 3rd-order predictor variable $\underline{\mathbf{X}}$ and 1st-order response variable y. Note that each block is computed using either ACE or ACCoS further referred to as ACE-BTTR and ACCoS-BTTR.

$$\mathbf{x}_r^{(n)} = (\underline{\mathbf{X}} \times_2 \mathbf{P}^{(2)T} \times_3 \ldots \times_n \mathbf{P}_{\backslash r}^{(n)T} \times_{n+1} \ldots \times_N \mathbf{P}^{(N)T})_{(1)} \mathrm{vec}(\underline{\mathbf{G}}_{(n)\backslash r}^{(c)}) \quad (2)$$

with $\underline{\mathbf{G}}_{(n)\backslash r}^{(c)} \in \mathbb{R}^{(R_n - 1) \times (R_1 \times \ldots \times R_N)}$ the n-mode matricization of the core tensor $\underline{\mathbf{G}}^{(c)}$ of which row r is removed. A lower score indicates a higher relevance of the associated (removed) component in the overall correlation between $\underline{\mathbf{X}}$ and $\mathbf{y}$.

In Step 2, we iteratively group the most relevant components. In order to select the smallest number of components that maximizes correlation, we start with the component with the lowest score in Step 1 and, then, iteratively add the one with the next lowest score. At each iteration, a new score is calculated using the R-squared test between $\mathbf{x}_{\mathrm{ind}}^{(n)}$ and $\mathbf{y}$ where

$$\mathbf{x}_{\mathrm{ind}}^{(n)} = (\underline{\mathbf{X}} \times_2 \mathbf{P}^{(2)T} \times_3 \ldots \times_n \mathbf{P}_{\mathrm{ind}}^{(n)T} \times_{n+1} \ldots \times_N \mathbf{P}^{(N)T})_{(1)} \mathrm{vec}(\underline{\mathbf{G}}_{(n)\mathrm{ind}}^{(c)}) \quad (3)$$

with ind the index of the first D components with lowest score and $\underline{\mathbf{G}}_{(n)\mathrm{ind}}^{(c)} \in \mathbb{R}^{D \times (R_n \times \ldots \times R_N)}$ the mode-n core tensor

```
Input: $\underline{\mathbf{X}} \in \mathbb{R}^{I_1 \times \ldots \times I_N}, \mathbf{y} \in \mathbb{R}^{I_1 \times 1}$
Output: $\underline{\mathbf{G}}^{(X)} \in \mathbb{R}^{1 \times R_2 \times \ldots \times R_N}, \mathbf{t}, \{\mathbf{P}^{(n)}\}_{n=2}^{N}$
 1: $\underline{\mathbf{G}}, \mathbf{t}, \{\mathbf{P}^{(n)}\}_{n=2}^{N}$ = $ACE(\underline{\mathbf{X}}, \mathbf{y})$
 2: select correlated components using Eq 2 and Eq 3
 3: $\mathbf{t} = (\underline{\mathbf{X}} \times_2 \mathbf{P}^{(2)T} \times_3 \ldots \times_N \mathbf{P}^{(N)T})_{(1)} vec(\underline{\mathbf{G}})$
 4: $\mathbf{t} = \mathbf{t}/\|\mathbf{t}\|_F$
 5: $\underline{\mathbf{G}}^{(X)} = [\![ \underline{\mathbf{X}}; \mathbf{t}^T, \mathbf{P}^{(2)T}, \ldots, \mathbf{P}^{(N)T} ]\!]$
 6: return $\underline{\mathbf{G}}^{(X)}, \mathbf{t}, \{\mathbf{P}^{(n)}\}_{n=2}^{N}$
```

Fig. 5. ACCoS

```
Input: $\underline{\mathbf{X}} \in \mathbb{R}^{I_1\times\dots\times I_N}, \mathbf{y} \in \mathbb{R}^{I_1\times 1}, K$
Output: $\{\mathbf{P}_k^{(n)}\}, \{\mathbf{t}_k\}, \underline{\mathbf{G}}_k^{(X)}$ for $k = 1, ..., K$ ; $n = 2, ..., N$
 1: Initialisation of $\underline{\mathbf{E}}_\mathbf{1} = \underline{\mathbf{X}}$ and $\mathbf{f}_\mathbf{1} = \mathbf{y}$
 2: for $k = 1$ to $K$ do
 3:     if $\|\underline{\mathbf{E}}_\mathbf{k}\| > \epsilon$ and $\|\mathbf{f}_\mathbf{k}\| > \epsilon$ then
 4:         $\underline{\mathbf{C}}_\mathbf{k} = \langle \underline{\mathbf{E}}_k, \mathbf{f}_k \rangle_{\{1,1\}}$
 5:         $\underline{\mathbf{G}}_k^{(X)}, \mathbf{t}_k, \mathbf{P}_k^{(2)}, ..., \mathbf{P}_k^{(N)}$ = $ACE(\underline{\mathbf{E}}_k, \mathbf{f}_k)$ or $ACCoS(\underline{\mathbf{E}}_k, \mathbf{f}_k)$
 6:         $\mathbf{b}_\mathbf{k} = \mathbf{t}_\mathbf{k}\mathbf{f}_\mathbf{k}$
 7:         $\underline{\mathbf{E}}_\mathbf{k+1} = \underline{\mathbf{E}}_\mathbf{k} - [\![\underline{\mathbf{G}}_k^{(X)}; \mathbf{t}_k, \mathbf{P}_k^{(2)}, ..., \mathbf{P}_k^{(N)}]\!]$
 8:         $\mathbf{f}_\mathbf{k+1} = \mathbf{f}_\mathbf{k} - \mathbf{t}_\mathbf{k}\mathbf{b}_\mathbf{k}$
 9:     else
10:         break
11:     end if
12: end for
```

Fig. 6. BTTR

$\underline{\mathbf{G}}^{(c)}$ in which D rows are selected. The selection process stops when the score starts to decrease. Finally, like ACE, given the new core tensor $\underline{\mathbf{G}}^{(c)}$ and factor matrices $\{\mathbf{P}^{(n)}\}_{n=2}^N$ with selected components, the score vector $\mathbf{t}$ is first computed as

$$\mathbf{t} = (\underline{\mathbf{C}} \times_2 \mathbf{P}^{(2)T} \times_3 ... \times_N \mathbf{P}^{(n)T})_{(1)}\text{vec}(\underline{\mathbf{G}}^{(c)})$$

and then normalized. The complete framework is summarized in Algorithm 5 and further referred to as ACCoS.

B. Block-Term Tensor Regression (BTTR)

We propose a novel Block-Term Regression (BTTR) model based on $(L_1^k, ..., L_N^k)$ BTD with automatic MTR determination. More specifically, BTTR is a deflation-based method in which the maximally correlated representations of $\underline{\mathbf{X}}$ and $\mathbf{y}$ are extracted via ACE/ACCoS at each iteration. Therefore, BTTR inherits the advantages of the proposed ACE/ACCoS and does not require one to set the model parameters manually. This provides BTTR with an additional important property: the ability to model complex data in which the optimal MTR is not necessarily stable across sequential decompositions. A scheme of BTTR is shown in Figure 2 whereas the full process is shown in Algorithm 6. In the rest of the paper, when required, we will use the terms ACE-BTTR and ACCoS-BTTR to refer to ACE and ACCoS in BTTR, respectively.

Given a set of training data $\underline{\mathbf{X}}_{\text{train}} \in \mathbb{R}^{I_1\times\dots\times I_N}$ and vectoral response $\mathbf{y}_{\text{train}} \in \mathbb{R}^{I_1}$, BTTR training consists of automatically identifying K blocks s.t.

$$\underline{\mathbf{X}}_{\text{train}} = \sum_{k=1}^{K} \underline{\mathbf{G}}_k \times_1 \mathbf{t}_k \times_2 \mathbf{P}_k^{(2)} \times_3 ... \times_N \mathbf{P}_k^{(n)} + \underline{\mathbf{E}}_k$$

$$\mathbf{y}_{\text{train}} = \sum_{k=1}^{K} \mathbf{u}_k + \mathbf{f}_k \text{ with } \mathbf{u}_k = \mathbf{t}_k b_k$$

with $\underline{\mathbf{G}}_k \in \mathbb{R}^{1\times R_2^k\times\dots\times R_N^k}$ the core tensor for the kth-block, $\mathbf{P}_k^{(n)}$ the kth loading matrix for the n-mode, $\mathbf{u}_k$ and $\mathbf{t}_k$ the score vectors, b_k the regression coefficient, and $\underline{\mathbf{E}}_k$ and $\mathbf{f}_k$ the residuals. Once the model is trained – and, hence, $\underline{\mathbf{G}}_k$, $\mathbf{P}_k^{(n)}$ and b_k are computed – the final prediction is obtained as: $\mathbf{y}_{\text{test}} = \mathbf{Tb} = \mathbf{X}_{\text{test}(1)}\mathbf{Wb}$ where each column $\mathbf{w}_k = (\mathbf{P}_k^{(n)} \otimes ... \otimes \mathbf{P}_k^{(2)})vec(\underline{\mathbf{G}}_k)$.

1) Generalized Block-Term Tensor Regression (gBTTR)

One possible limitation of BTTR is the lack of generalization as it aims to maximize the correlation between $\underline{\mathbf{X}}_{\text{train}}$ and $\mathbf{y}_{\text{train}}$ only, and this can lead to overfitting. Starting from the above prediction equation, we can re-write the above as the well-known linear regression equation $\mathbf{y}_{\text{test}} = \mathbf{X}_{\text{test}(1)}\mathbf{b}_{\text{BTTR}}$ with $\mathbf{b}_{\text{BTTR}} = \mathbf{Wb}$.

In the context of linear regression, it has been shown that by randomly adding Gaussian noise $\epsilon \sim (0, \sigma)$ to the inputs increases the robustness of the model against overfitting while improving its generalization [48]. More specifically, the process consists of fitting the linear regression to the manipulated noisy data. Clearly, different regression lines are obtained each time random noise is added to the data. Therefore, to obtain a stable result, a process called marginalization is introduced to integrate out such randomness. In practice, marginalization consists of averaging the estimated regression lines. Such averaged regression line with noise σ is equal to the ridge regression line with penalty parameter $\lambda = N\sigma^2$ [48] with N the number of observations in the data set. We adopt the aforementioned idea as follows: At each iteration k, $\mathbf{f}_k$ is manipulated M times by adding random Gaussian noise; for each $\mathbf{f}_k^M$ the corresponding $\mathbf{w}_\mathbf{k}^\mathbf{M}\mathbf{b}_\mathbf{k}^\mathbf{M}$ solution is determined; then, we compute the averaged solution (marginalization), $\hat{\mathbf{w}}_\mathbf{k}\hat{\mathbf{b}}_\mathbf{k}$, followed by deflation before the next iteration. The final prediction is $\mathbf{y}_{\text{test}} = \mathbf{X}_{\text{test}(1)}\hat{\mathbf{b}}_{\text{BTTR}}$ where the kth element in $\hat{\mathbf{b}}_{\text{BTTR}}$ is computed via marginalization in the kth iteration, i.e. $\hat{\mathbf{w}}_\mathbf{k}\hat{\mathbf{b}}_\mathbf{k}$. We will refer to this ridge-like approach as generalized BTTR (gBTTR).

C. Time Complexity Analysis

Given training data $\underline{\mathbf{X}}_{\text{train}} \in \mathbb{R}^{I_1\times\dots\times I_N}$ and vectoral response $\mathbf{y}_{\text{train}} \in \mathbb{R}^{I_1}$ with $I_1 = I_2 = ... = I_N$ and $R_1 = R_2 = ... = R_N$, gBTTR has appromimately time complexity $O(I^{(N)}R + R^{3(N-1)} + R^{2(N-1)}I)$. For comparison, HOPLS, under the same circumstances, has time complexity $O(I^{(N+M)}R^2 + R^{4(N+M-2)} + R^{3(N+M-2)}I)$ [49].

D. Non-multilinear approaches

As multilinear approaches are less common in BCI, we will consider the BCI competition IV dataset (see further) for comparing the performance of BTTR and HOPLS with the winner of this competition a linear regression model based on amplitude modulation (AM) [22], as well as with more recent attempts based on Random Forests (RF), Convolutional Neural Networks (CNN), and Long Short-Term Memory Network (LSTM) (proposed and compared in [23]).

Since in both [22] and [23] only 1 test set was used to assess performance, which factually impedes statistically testing the significance of any observed performance difference between the compared algorithms, we have re-implemented AM, RF, LARS, CNN and LSTM. We proceeded as follows.

For AM, we replicated the procedure described in [22]. First, we filtered the ECoG signals in the sub-gamma (1-60Hz), gamma (60-100 Hz) and high-gamma bands (100-200 Hz). For each of these bands, we determined the amplitude modulation and used it to estimate their band-specific AM features. For each finger and subject, we used forward feature selection using a wrapper approach to find the relevant AM features. These features are then used in a linear regression model. Finally, we verified whether the obtained performance compared with the one reported in [22].

For LARS, we used the *LassoLars* function from the most recent version of the *scikit-learn* package of Python (version $0.24.2$ released in April 2021; we assume that Xie et al. used version $0.19.1$, but this was not reported). Hence, small differences in performance could be occur. LARS transforms the original signal using ICA, decomposes it into different bands, calculates band powers and fits a LassoLars model. LassoLars has one main parameter, α, the multiplier for the penalty term whereby $\alpha = 0$ corresponds to ordinary least square linear regression. A line search was used to optimize $alpha$.

For RF, a similar processing pipeline was used. The *RandomForestRegressor* function from the *scikit-learn* package was used (same version as above). RF transforms the original signal using ICA, decomposes it into different bands, calculates band powers and fits the *RandomForestRegressor* model.

The CNN and LSTM were built according to the specifications and architecture given in [23]. CNN refers to a linear regression model applied to features prior extracted using a CNN. The CNN and LSTM were build using PyTorch in Python (version 1.8.0 released in March 2021; note that Xie et al. did not specify which version they used).

Temporal SVM-rbf and Temporal lightGBM, introduced in [27] use Riemannian-space features and temporal dynamics of the ECoG signal combined with modern machine learning.

IV. Data Sets Used

We first use synthetic data to compare state-of-the-art methods of automatic MTR determination versus the proposed ACE and ACCoS. Then, we adopt a real-world scenario for BTTR, HOPLS and several linear regression- and deep learning methods: decoding finger movement trajectories from ECoG recordings in humans, a challenging goal in the BCI community due to the presence of irrelevant and nonstationary brain activity and noise.

A. Synthetic Data For Component Analysis

The synthetic data is created for the predictor Nth-order tensor variable $\underline{\mathbf{X}} = \underline{\mathbf{X}}_c + (1-\alpha)\underline{\mathbf{X}}_u \in \mathbb{R}^{I_1 \times \ldots \times I_N}$ and the response vector variable $\mathbf{y} \in \mathbb{R}^{I_1 \times 1}$. The response variable $\mathbf{y}$ is randomly generated. The $\underline{\mathbf{X}}_u \sim \mathcal{N}(0,1)$ is an uncorrelated tensor with randomly generated entries while $\underline{\mathbf{X}}_c$ is the true correlated tensor, defined according to the Tucker model as $\underline{\mathbf{X}}_c = [\![\underline{\mathbf{G}}, \mathbf{t}, \mathbf{P}^{(2)}, ..., \mathbf{P}^{(N)}]\!]$ with $\mathbf{t}, \mathbf{P}$ the score vector and the factor matrices, and $\underline{\mathbf{G}} \sim \mathcal{N}(0,1) \in \mathbb{R}^{1 \times R_2 \times \ldots \times R_N}$ the randomly generated core tensor. Note than $R_1 = 1$. The entries (r_n, i_n) of the loading factor $\mathbf{P}^{(n)} \in \mathbb{R}^{I_n \times R_n}$ are defined as

$$\mathbf{P}_r^{(n)} \begin{cases} sin(2\pi r_2 \frac{i_2}{I_2}) \text{ for } n = 2 \\ cos(2\pi r_3 \frac{i_3}{I_3}) \text{ for } n = 3 \\ sgn(sin(2\pi r_4 \frac{i_4}{I_4})) \text{ for } n = 4 \end{cases}$$

with $r_n = 1, ..., R_n$ and $i_n = 1, ..., I_n$, whereas the score factor $t \in \mathbb{R}^{I_1 \times 1}$ is set equal to $\mathbf{y}$. The parameter α defines the weight of $\underline{\mathbf{X}}_u$ on the final tensor $\underline{\mathbf{X}}$ and, therefore, regularizes the correlation level between $\underline{\mathbf{X}}$ and $\mathbf{y}$ with $\alpha = 1$ and $\alpha = 0$ yielding, respectively, maximal and minimal correlation.

We performed a series of tests to evaluate the robustness of the proposed ACE model as well as mPSTD and ACCoS under different conditions and compared their performances with the state-of-the-art models in automatic rank determination (see Previous work on automatic multilinear tensor rank selection). Several tests are performed by varying data properties aimed at showing how they affect the recovery of the true correlated (hidden) components. We chose N=4 and varied one property at a time while keeping the others constant. The data properties we explored are listed below:

- Correlation level: parameter α is chosen as 0.1, 0.5 or 1 to regularize the interference of uncorrelated sources.
- Number of components (i.e. MTR): parameter set $R_2, ..., R_N$ defines the size of the (correlated) factors $\mathbf{P}^{(n)} \in \mathbb{R}^{I_n \times R_N}{}_{n=2}^{N}$, i.e., the components of interest, and are heuristically chosen as $[1,1,1]$, $[2,2,1]$, $[2,1,2]$, $[1,2,2]$, or $[2,2,2]$.
- Tensor dimensionality: parameter set $I_2, ..., I_N \in [5,5,5], [10,10,10]$, or $[20,20,20]$ defines the dimensionality of the Nth-order independent variable $\underline{\mathbf{X}}$.
- Sample size: parameter $I_1 \in 100, 1000,$ or 10000 defines the number of observations.

We used two values to assess performance: how accurately the expected true components are retrieved (C1) and how much of the relevant information is contained in the extra components (C2). Note that we report NA for C2 when no extra components are available. We repeated each test 10 times and for each run the data was randomly generated as described above. Finally, we reported C1 and C2 in terms of mean and standard deviation as well as number of extra components averaged across modes.

B. Regression Analysis Of Real-Word Data: Decoding Finger Movement Trajectories From ECoG Signals Recorded in Humans

We also compared the proposed BTTR versions with HOPLS as well as several conventional linear regression approaches and deep learning networks in a real-word application. We adopted the dataset used in BCI competition IV where the task was to predict continuous finger flexions from ECoG signals recorded from the motor cortex, sampled at 1000 Hz. Three subjects were cued to move a particular finger at a particular moment (gauged with a data glove). In total, 150 trials were executed (30 trials per finger) in a single session lasting 600 seconds. For each trial, subjects typically flexed the cued finger 3–5 times for 2 seconds followed by a rest period of 2 seconds. The first 400 and the last 200 seconds of recording were used as training and testing set respectively. The number of ECoG channels varied across subjects. More details about the data can be found in [36].

1) Data preparation for multiway analysis

First, the ECoG and data glove signals were preprocessed. The data glove data was, independently for each finger, normalized (z-scores) yielding a vector $y \in \mathbb{R}^{\text{Samples}}$ for each finger. The ECoG signal was first inspected to identify and further exclude bad channels: we found that channels 55 in subject 1, 21 and 38 in subject 2, and 50 in subject 3 were affected by strong artifacts and therefore removed. The remaining channels were re-referenced using the common average reference (CAR) technique [50]. Then, using bidirectional fourth-order Butterworth band-pass filters, the ECoG signals were subjected to 8 band-pass filters, as commonly done in ECoG based BCI [37]: δ (1.5 -5 Hz), θ (5 -8 Hz), α (8 -12 Hz), β_1 (12 -24 Hz), β_2 (24 -34 Hz), γ_1 (34 -60 Hz), γ_2 (60 - 100 Hz), γ_3 (100 - 130 Hz). For each cued finger flexion trial, the glove data and the bandpass filtered ECoG signals were extracted starting 1 sec prior to trial onset ("epoch"). ECoG epochs were downsampled to 10 Hz to further reduce data size. All epochs were then concatenated into a unique fourth-order tensor $\underline{\mathbf{X}} \in \mathbb{R}^{\text{Samples} \times \text{Channels} \times 8\text{(Frequencies)} \times 10\text{(Time)}}$. There were 62, 48, and 64 ECoG channels for subjects 1, 2, and 3, respectively. Finally, $\underline{\mathbf{X}}$ is normalized (z-scores) to reduce the difference in magnitude between frequency bands. A graphical representation of the data preparation is shown in Figure 7.

2) Calibration, validation, performance assessment

A 5-fold cross-validation approach was used to optimize the model parameters on the training data, i.e., $K, R_2, ..., R_N$ for HOPLS and K for BTTR. In addition, in support of the statistical analysis, the test data was split into 5 non-overlapping blocks. We also proceed in this way for the winning algorithm of the BCI competition IV, as well as its more recent attempts.

The decoding performance for each block was measured using Pearson's correlation coefficient between predicted and actual movement trajectory, which are then averaged. We used a two-tailed Wilcoxon signed-rank test [51] to compare the models for each subject. Two results are considered significantly different if the p-value is < 0.05.

V. Results

In this section, we compare the results obtained for the state-of- the-art methods of automatic MTR selection and the proposed ACE and ACCoS for the case of synthetic data, under different conditions. Then we compare the predictive performance of the proposed multiway regression, BTTR, with and without automatic component selection, for a real-world scenario.

A. ACE And ACCoS On Synthetic Data For Component Extraction

For the synthetic data case, we performed various tests to verify model robustness under different conditions such as 1) correlation level between $\underline{\mathbf{X}}$ and $\mathbf{y}$, 2) rank size, 3) tensor dimensionality, and 4) sample size. We first show the efficiency of the proposed ACE against manual tuning of model parameters required for the proposed mPSTD and standard PSTD. Then we compare ACE and ACCoS with state-of-the-art methods under the same conditions.

In Figure 8 we show an example of mPSTD and standard PSTD (top panels) as well the outcome of the BIC analysis included in ACE for the automatic selection of model parameters SNR and τ (bottom panels) when $\alpha = 0.5$, $R_2, R_3, R_4 = [2, 2, 2]$, $I_1 = 10000$, and $I_2, I_3, I_4 = [20, 20, 20]$. Initialization of the factor matrices seems to be a key element in extracting the correct components and in overall convergence. This is not noticed when comparing mPSTD and standard PSTD (Figure 8, top panel). The proposed mPSTD tends to be less sensitive to the choice of model parameters as for standard PSTD, which relies on random initialization, only few SNR-τ combinations lead to a correct solution. Interestingly, mPSTD generally yields a more accurate solution (max $C1 = 1$) than standard PSTD (max $C1 = 0.81$). In addition, the outcome of this study shows that ACE, as it relies on the BIC approach, is able to determine the optimal model parameters (leading to $C1 = 1$) under any condition in a fully automatic way. An obvious advantage of the proposed ACE is the automatic selection of the parameters –generally obtained via cross-validation– and the absence of any prior assumptions.

Additional tests showed that both ACCoS and ACE perform at least as good as the other state-of-the-art methods TREL1 and SCORE as they are able to retrieve, in a fully automatic way, the true correlated components regardless of the properties of the data (cf., C1 result). However, ACCoS differs from the others as it can successfully identify and reject irrelevant components (cf., C2 result). An example is reported in Figure 9 where the correlation level between the two variables changes while fixing $R_2, R_3, R_4 = [2, 2, 2]$, $I_1 = 10000$, and $I_2, I_3, I_4 = [20, 20, 20]$.

B. BTTR on Real-World Data: Decoding Finger Movement Trajectories from Human ECoG Recordings

For clarity's sake, we first compared the proposed versions of BTTR against HOPLS to select the best BTTR version on a real-world case,then compared this version against the BCI competition IV winner and the more recent attempts. We analyzed for each subject one finger at a time. Figure 10 shows an example of predicted finger movement for gBTTR and HOPLS together with the data glove signal.

1) Comparison of multiway regression approaches: BTTR versions against HOPLS

In Figure 11, we show for each subject (panels shown row-wise) the averaged (across fingers) correlation coefficients obtained when predicting test samples (Study 1, panels in left column) and when predicting samples used for model training (Study 2, panels in right column).

When analyzing the results of Study 1, statistical analysis (see Calibration, validation, performance assessment) revealed no significant difference between the performance of BTTR and that of its generalized version gBTTR. However, ACCoS-

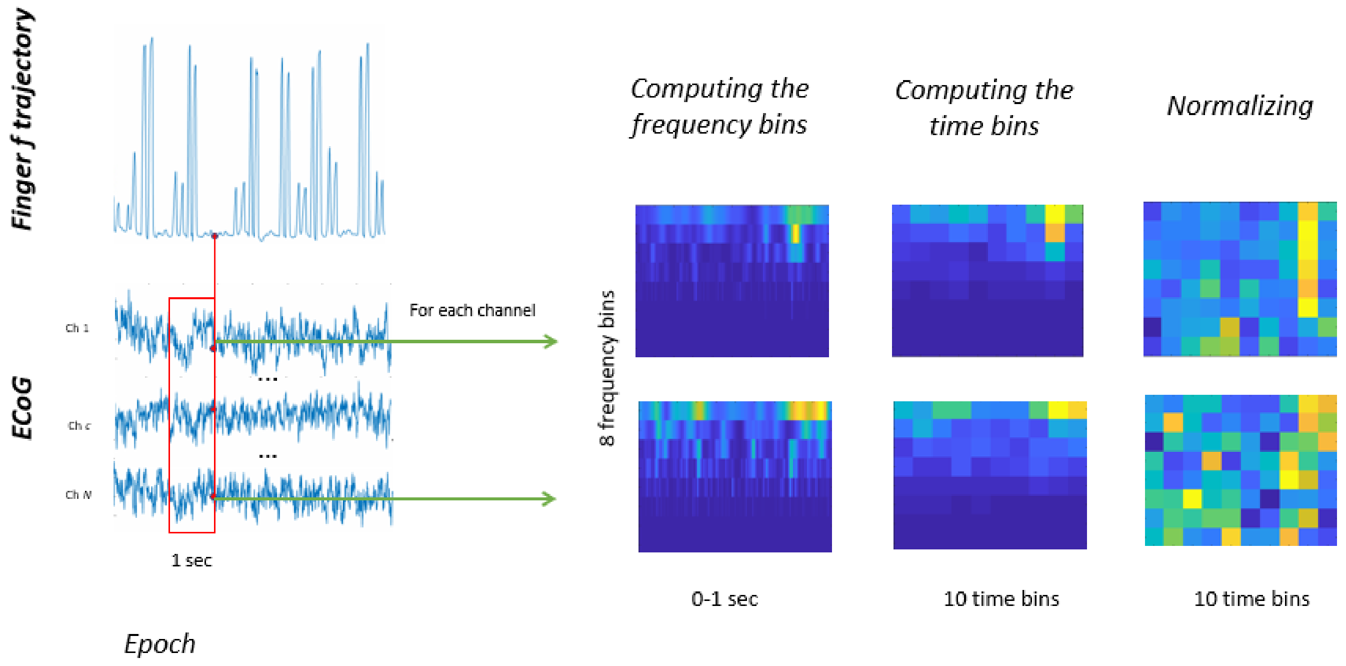


Fig. 7. Diagram with the steps to create an ECoG tensor.

gBTTR performs better compared to ACE-gBTTR and HOPLS for subject 3 (gray brackets).

In Study 2, ACCoS-BTTR performs equally well as ACE-BTTR and both models provide better accuracies than HOPLS (black brackets, all subjects). When considering the gBTTR version, performances are lower than with BTTR. Indeed, except for subject 2 (gray brackets), the difference with HOPLS tends to disappear when using gBTTR.

To summarize, for ACCoS-gBTTR, prediction accuracy of the test data improved while that of its training data decreased: the predictions obtained with ACCoS-BTTR, ACE-BTTR and ACE-gBTTR are comparable to the ones obtained with HOPLS while for ACCoS-gBTTR they are at least as good. Hence, seemingly, the generalized version of BTTR is able to limit the effect of overfitting mainly for ACCoS-BTTR; indeed, ACE-BTTR extracts also components that are not of interest and that -as in the HOPLS case–demote prediction accuracy.

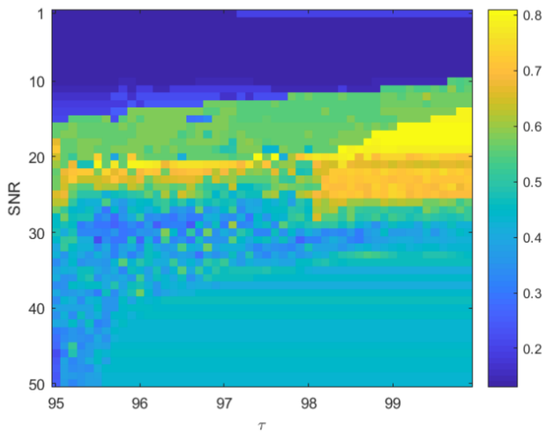
We further investigated the runtime required by these models to identify their parameters. Overall, ACE-BTTR and ACCoS-BTTR are trained faster than HOPLS (e.g. 3 minutes and 8 minutes respectively against 14 hours for HOPLS). This is in line with the time complexity found for the algorithms. Note that HOPLS requires computationally expensive techniques such as cross-validation to identify the optimal set of model parameters (i.e., the number of scores and loadings). In contrast, BTTR and gBTTR automatically determine the model parameters (i.e., the number of loadings) leading to a model that combines high flexibility with more natural representation of complex multiway data, although it still requires (a simple) cross-validation to determine the number of scores. The next comparison continues with ACE-BTTR as it is the most well-rounded of the various versions in terms of speed and performance.

2) Comparison with non-multiway approaches

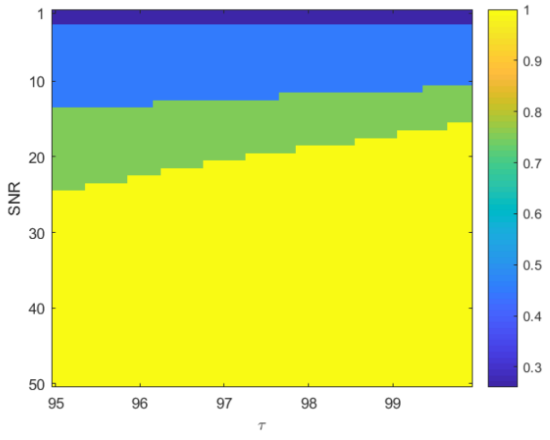
We also compared the ACE-BTTR model and HOPLS with the winner of the BCI competition IV, linear regression of amplitude modulation (AM) [22], as well as 4 more recent attempts as the competition concluded some time ago: Random Forest (RF), Least Angle Regression (LARS), Convolutional Neural Network (CNN), and Long Short-Term Memory Network (LSTM). The correlation coefficients are given for each finger individually and averaged across all fingers, except for finger 4 (ring) as flexing the latter is difficult to suppress when the 3rd or 5th finger is flexing. The average results and their standard deviations for the 5 blocks of test data (see above) are listed in Table II, Table III and Table IV (listed under Avg.) for subjects 1, 2 and 3, respectively.

For all 3 subjects, a statistically significant difference (see Calibration, validation, performance assessment) in average results was found between ACE-BTTR and LSTM and between AM/RF/LARS and ACE-BTTR. It is interesting to note that on a per-finger basis, there was not always a statistically significant difference between individual algorithms. For instance, for all 3 subjects, there is no statistical difference for the thumb and index finger between ACE-BTTR and LSTM. For subjects 1 and 3, there is a significant difference for the middle finger between ACE-BTTR and LSTM. For subjects 1 and 2, there is a significant difference for the pinky between ACE-BTTR and LSTM.

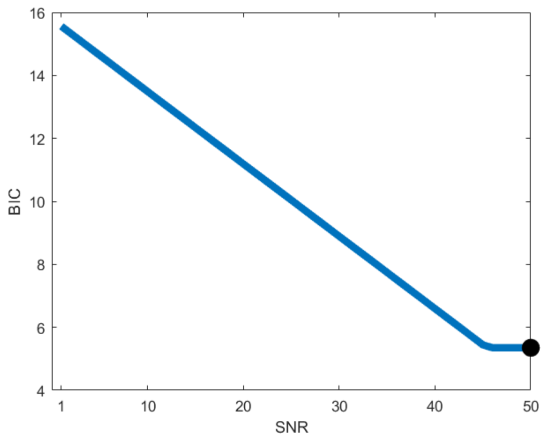
SVM-rbf and LightGBM [27] are the main state-of-the-art methodologies. ACE-BTTR performs better for subject 1 compared to these methodologies. LightGBM performs better compared to ACE-BTTR for subject 2. Looking at a per-finger case, the main difference is the middle finger which performs far better in LightGBM. For subject 3, ACE-BTTR performs



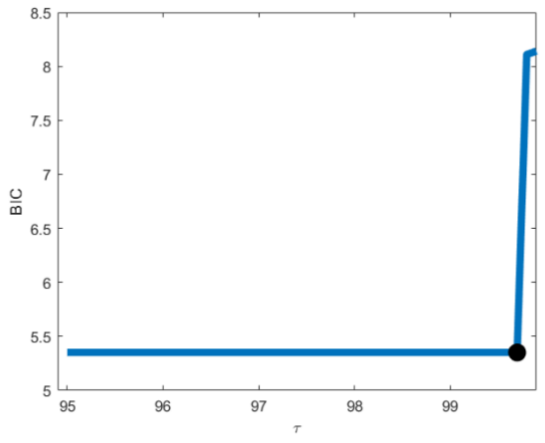
(a) a



(b) b

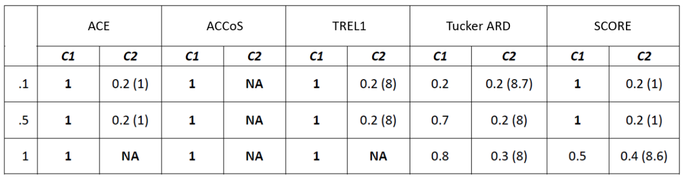


(c) c



(d) d

Fig. 8. Top left panel (Standard PSTD): C1 accuracy (color coded, see scale on the right) achieved by standard PSTD when (manually) varying model parameters SNR and τ. Top right panel (mPSTD): idem but for the proposed mPSTD. Bottom left (BIC-SNR) and right (BIC-Ratio for best SNR): filled circles indicate the selected SNR (50) and τ (99.7) values.



	ACE		ACCoS		TREL1		Tucker ARD		SCORE	
	C1	C2	C1	C2	C1	C2	C1	C2	C1	C2
.1	1	0.2 (1)	1	NA	1	0.2 (8)	0.2	0.2 (8.7)	1	0.2 (1)
.5	1	0.2 (1)	1	NA	1	0.2 (8)	0.7	0.2 (8)	1	0.2 (1)
1	1	NA	1	NA	1	NA	0.8	0.3 (8)	0.5	0.4 (8.6)

Fig. 9. Results of correlation analysis when varying the correlation parameter α while keeping $R_2, R_3, R_4 = [2, 2, 2]$, $I_1 = 10000$, and $I_2, I_3, I_4 = [20, 20, 20]$. The reported values represent the correctness of the expected components (C1) and the extra components (C2 with average of number of extra components in brackets).

Methods	Thumb	Index	Middle	Ring	Pinky	Avg.
ACE-BTTR	0.72 ± .06	0.77 ± .09	0.38 ± .02	0.68 ± .04	0.67 ± .02	0.64 ± .05
HOPLS	0.70 ± .05	0.79 ± .08	0.36 ± .03	0.70 ± .06	0.65 ± .02	0.63 ± .04
AM	0.57 ± .03	0.69 ± .06	0.14 ± .02	0.52 ± .04	0.28 ± .01	0.42 ± .03
RF	0.58 ± .09	0.54 ± .05	0.07 ± .03	0.31 ± .05	0.33 ± .02	0.38 ± .05
LARS	0.11 ± .05	0.08 ± .03	0.10 ± .02	0.60 ± .05	0.39 ± .02	0.17 ± .03
CNN	0.67 ± .04	0.78 ± .04	0.11 ± .02	0.54 ± .03	0.45 ± .04	0.50 ± .04
LSTM	0.73 ± .03	0.79 ± .08	0.18 ± .02	0.61 ± .04	0.45 ± .04	0.54 ± .04
SVM-rbf	0.594	0.734	0.364	0.395	0.480	0.54
LightGBM	0.543	0.760	0.401	0.383	0.531	0.558

TABLE II
FINGER TRAJECTORY DECODING PERFORMANCE (PEARSON CORRELATION) OF BTTR AND OTHER MODELS FOR SUBJECT 1.

equally to LightGBM with better decoding for middle, ring and thumb, but worse performance for the thumb and index finger.

Methods	Thumb	Index	Middle	Ring	Pinky	Avg.
ACE-BTTR	0.64 ± .05	0.46 ± .08	0.27 ± .04	0.50 ± .03	0.48 ± .01	0.46 ± .05
HOPLS	0.63 ± .04	0.47 ± .06	0.26 ± .05	0.51 ± .02	0.48 ± .01	0.44 ± .04
AM	0.52 ± .03	0.36 ± .06	0.23 ± .02	0.48 ± .04	0.33 ± .01	0.36 ± .03
RF	0.52 ± .05	0.36 ± .04	0.22 ± .03	0.39 ± .04	0.25 ± .02	0.34 ± .04
LARS	0.54 ± .05	0.41 ± .04	0.18 ± .02	0.44 ± .04	0.25 ± .02	0.35 ± .03
CNN	0.60 ± .04	0.40 ± .04	0.24 ± .02	0.44 ± .03	0.28 ± .04	0.38 ± .04
LSTM	0.62 ± .03	0.38 ± .08	0.27 ± .02	0.47 ± .04	0.30 ± .04	0.39 ± .04
SVM-rbf	0.591	0.481	0.344	0.421	0.437	0.463
LightGBM	0.658	0.522	0.387	0.392	0.407	0.493

TABLE III
IDEM TO TABLE II BUT FOR SUBJECT 2.

VI. CONCLUSION

Tensor techniques can be an important asset to BCI as they can outperform more conventional decoders in accuracy and

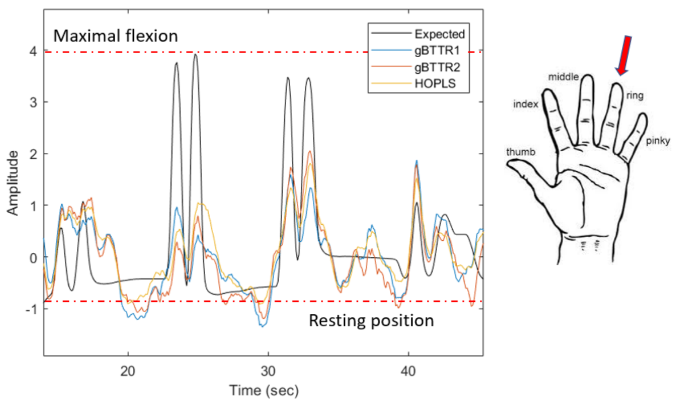


Fig. 10. Predicted ring finger flexions of Subject 1. Actual (data glove) and predicted (regression models) finger flexion amplitudes (z-scores) are plotted as function of time (in ms), Curve colors are explained in the inset. For the sake of exposition, only results for the generalized version gBTTR are shown.

Methods	Thumb	Index	Middle	Ring	Pinky	Avg.
ACE-BTTR	0.73 ± .05	0.59 ± .08	0.64 ± .04	0.63 ± .02	0.72 ± .01	0.67 ± .05
HOPLS	0.74 ± .06	0.57 ± .09	0.65 ± .02	0.61 ± .04	0.68 ± .02	0.64 ± .04
AM	0.59 ± .03	0.51 ± .06	0.32 ± .02	0.53 ± .04	0.42 ± .01	0.46 ± .03
RF	0.67 ± .05	0.27 ± .04	0.16 ± .03	0.14 ± .04	0.36 ± .02	0.37 ± .04
LARS	0.72 ± .05	0.43 ± .04	0.45 ± .02	0.51 ± .04	0.64 ± .02	0.56 ± .03
CNN	0.74 ± .03	0.53 ± .05	0.45 ± .04	0.49 ± .03	0.68 ± .06	0.60 ± .05
LSTM	0.74 ± .02	0.55 ± .06	0.46 ± .04	0.41 ± .02	0.75 ± .06	0.62 ± .05
SVM-rbf	0.767	0.654	0.513	0.347	0.638	0.643
LightGBM	0.82	0.648	0.578	0.358	0.664	0.6775

TABLE IV
IDEM TO TABLE II BUT FOR SUBJECT 3.

reliability but their proliferation is hindered by the computational intensive parameter estimation. Model parameters are often optimized via time consuming techniques such as cross-validation on a sufficient set of parameter combinations. To tackle this limitation, we proposed a new tensor decomposition approach for regression that we enhanced with automatic rank determination and showed that it can challenge state-of-the-art multiway regression techniques while outperforming more conventional ones. The proposed solution is characterized by flexible modeling, supporting the representation of complex data, and by fast model training. This can open new perspectives for multiway data modeling in BCI applications.

ACKNOWLEDGMENT

AF was supported by the National Fund for Scientific Research – Flanders (FWO 1157019N). FC was supported by the Hermes Fund of the National Fund for Scientific Research – Flanders (IWT.141112). MMVH is supported by research grants received from the European Union's Horizon 2020 research and innovation programme under grant agreement No. 857375, the special research fund of the KU Leuven (C24/18/098), the Belgian Fund for Scientific Research – Flanders (G0A0914N, G0A4118N, G0A4321N), the Interuniversity Attraction Poles Programme — Belgian Science Policy (IUAP P7/11), and the Hercules Foundation (AKUL 043).

REFERENCES

[1] A. Vallabhaneni, T. Wang, and B. He, "Brain—computer interface," in *Neural engineering*. Springer, 2005, pp. 85–121.

[2] A. B. Ajiboye, F. R. Willett, D. R. Young, W. D. Memberg, B. A. Murphy, J. P. Miller, B. L. Walter, J. A. Sweet, H. A. Hoyen, M. W. Keith *et al.*, "Restoration of reaching and grasping movements through brain-controlled muscle stimulation in a person with tetraplegia: a proof-of-concept demonstration," *The Lancet*, vol. 389, no. 10081, pp. 1821–1830, 2017.

[3] C. E. Bouton, A. Shaikhouni, N. V. Annetta, M. A. Bockbrader, D. A. Friedenberg, D. M. Nielson, G. Sharma, P. B. Sederberg, B. C. Glenn, W. J. Mysiw *et al.*, "Restoring cortical control of functional movement in a human with quadriplegia," *Nature*, vol. 533, no. 7602, pp. 247–250, 2016.

[4] L. R. Hochberg, M. D. Serruya, G. M. Friehs, J. A. Mukand, M. Saleh, A. H. Caplan, A. Branner, D. Chen, R. D. Penn, and J. P. Donoghue, "Neuronal ensemble control of prosthetic devices by a human with tetraplegia," *Nature*, vol. 442, no. 7099, pp. 164–171, 2006.

[5] J. L. Collinger, B. Wodlinger, J. E. Downey, W. Wang, E. C. Tyler-Kabara, D. J. Weber, A. J. McMorland, M. Velliste, M. L. Boninger, and A. B. Schwartz, "High-performance neuroprosthetic control by an individual with tetraplegia," *The Lancet*, vol. 381, no. 9866, pp. 557–564, 2013.

[6] L. R. Hochberg, D. Bacher, B. Jarosiewicz, N. Y. Masse, J. D. Simeral, J. Vogel, S. Haddadin, J. Liu, S. S. Cash, P. Van Der Smagt *et al.*, "Reach and grasp by people with tetraplegia using a neurally controlled robotic arm," *Nature*, vol. 485, no. 7398, pp. 372–375, 2012.

[7] B. Wodlinger, J. Downey, E. Tyler-Kabara, A. Schwartz, M. Boninger, and J. Collinger, "Ten-dimensional anthropomorphic arm control in a human brain- machine interface: difficulties, solutions, and limitations," *Journal of neural engineering*, vol. 12, no. 1, p. 016011, 2014.

[8] W. Wang, J. L. Collinger, A. D. Degenhart, E. C. Tyler-Kabara, A. B. Schwartz, D. W. Moran, D. J. Weber, B. Wodlinger, R. K. Vinjamuri, R. C. Ashmore *et al.*, "An electrocorticographic brain interface in an individual with tetraplegia," *PloS one*, vol. 8, no. 2, p. e55344, 2013.

[9] E. S. Nurse, S. E. John, D. R. Freestone, T. J. Oxley, H. Ung, S. F. Berkovic, T. J. O'Brien, M. J. Cook, and D. B. Grayden, "Consistency of long-term subdural electrocorticography in humans," *IEEE Transactions on Biomedical Engineering*, vol. 65, no. 2, pp. 344–352, 2017.

[10] K. J. Miller, L. B. Sorensen, J. G. Ojemann, and M. Den Nijs, "Power-law scaling in the brain surface electric potential," *PLoS Comput Biol*, vol. 5, no. 12, p. e1000609, 2009.

[11] R. J. Staba, C. L. Wilson, A. Bragin, I. Fried, and J. Engel Jr, "Quantitative analysis of high-frequency oscillations (80–500 hz) recorded in human epileptic hippocampus and entorhinal cortex," *Journal of neurophysiology*, vol. 88, no. 4, pp. 1743–1752, 2002.

[12] Z. C. Chao, Y. Nagasaka, and N. Fujii, "Long-term asynchronous decoding of arm motion using electrocorticographic signals in monkey," *Frontiers in neuroengineering*, vol. 3, p. 3, 2010.

[13] T. Ball, M. Kern, I. Mutschler, A. Aertsen, and A. Schulze-Bonhage, "Signal quality of simultaneously recorded invasive and non-invasive eeg," *Neuroimage*, vol. 46, no. 3, pp. 708–716, 2009.

[14] J. C. Siero, D. Hermes, H. Hoogduin, P. R. Luijten, N. F. Ramsey, and N. Petridou, "Bold matches neuronal activity at the mm scale: A combined 7 t fmri and ecog study in human sensorimotor cortex," *Neuroimage*, vol. 101, pp. 177–184, 2014.

[15] G. Schalk, K. J. Miller, N. R. Anderson, J. A. Wilson, M. D. Smyth, J. G. Ojemann, D. W. Moran, J. R. Wolpaw, and E. C. Leuthardt, "Two-dimensional movement control using electrocorticographic signals in humans," *Journal of neural engineering*, vol. 5, no. 1, p. 75, 2008.

[16] T. Yanagisawa, M. Hirata, Y. Saitoh, H. Kishima, K. Matsushita, T. Goto, R. Fukuma, H. Yokoi, Y. Kamitani, and T. Yoshimine, "Electrocorticographic control of a prosthetic arm in paralyzed patients," *Annals of neurology*, vol. 71, no. 3, pp. 353–361, 2012.

[17] E. A. Felton, J. A. Wilson, J. C. Williams, and P. C. Garell, "Electrocorticographically controlled brain–computer interfaces using motor and sensory imagery in patients with temporary subdural electrode implants: report of four cases," *Journal of neurosurgery*, vol. 106, no. 3, pp. 495–500, 2007.

[18] A. L. Benabid, T. Costecalde, A. Eliseyev, G. Charvet, A. Verney, S. Karakas, M. Foerster, A. Lambert, B. Morinière, N. Abroug *et al.*, "An exoskeleton controlled by an epidural wireless brain–machine interface in a tetraplegic patient: a proof-of-concept demonstration," *The Lancet Neurology*, vol. 18, no. 12, pp. 1112–1122, 2019.

[19] J. Kubanek, K. J. Miller, J. G. Ojemann, J. R. Wolpaw, and G. Schalk, "Decoding flexion of individual fingers using electrocorticographic signals in humans," *Journal of neural engineering*, vol. 6, no. 6, p. 066001, 2009.

[20] Z. Wang, Q. Ji, K. J. Miller, and G. Schalk, "Decoding finger flexion from electrocorticographic signals using a sparse gaussian process," in

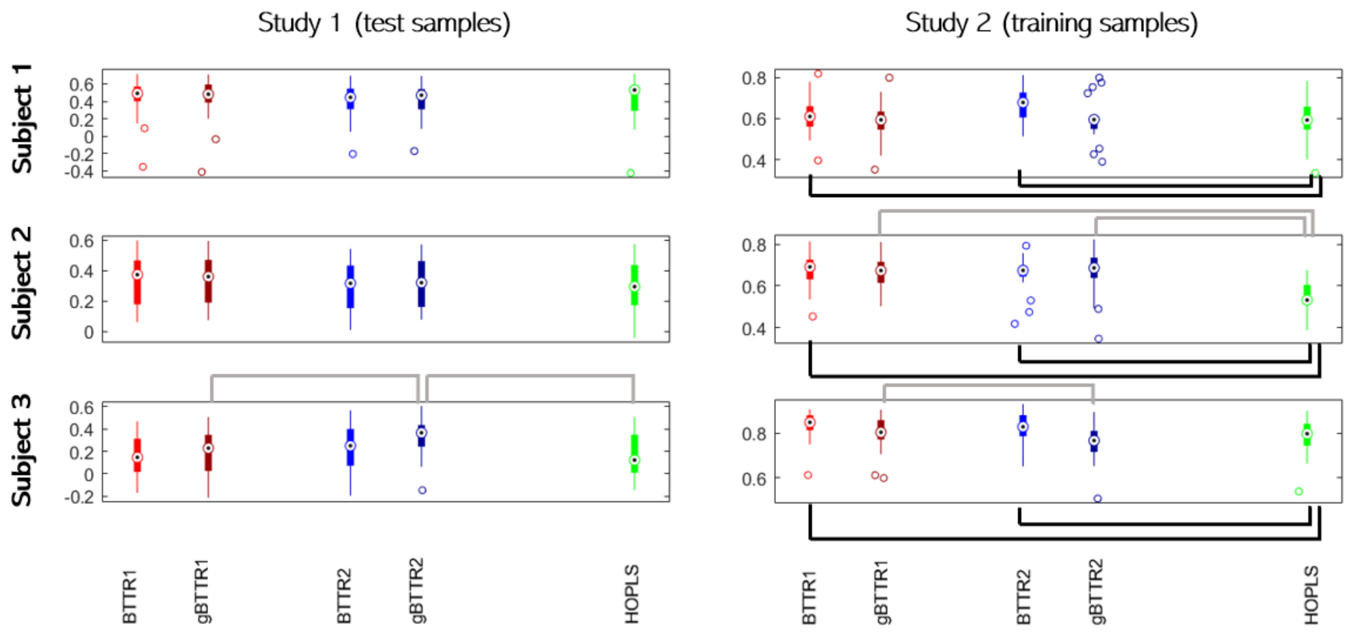


Fig. 11. Performances of the proposed models and HOPLS when predicting training and test data.

2010 20th International conference on pattern recognition. IEEE, 2010, pp. 3756–3759.

[21] R. Flamary and A. Rakotomamonjy, “Decoding finger movements from ecog signals using switching linear models,” *Frontiers in neuroscience*, vol. 6, p. 29, 2012.

[22] N. Liang and L. Bougrain, “Decoding finger flexion from band-specific ecog signals in humans,” *Frontiers in neuroscience*, vol. 6, p. 91, 2012.

[23] Z. Xie, O. Schwartz, and A. Prasad, “Decoding of finger trajectory from ecog using deep learning,” *Journal of neural engineering*, vol. 15, no. 3, p. 036009, 2018.

[24] G. Pfurtscheller and C. Neuper, “Motor imagery and direct brain-computer communication,” *Proceedings of the IEEE*, vol. 89, no. 7, pp. 1123–1134, 2001.

[25] E. Acar and B. Yener, “Unsupervised multiway data analysis: A literature survey,” *IEEE transactions on knowledge and data engineering*, vol. 21, no. 1, pp. 6–20, 2008.

[26] A. Smilde, R. Bro, and P. Geladi, *Multi-way analysis: applications in the chemical sciences*. John Wiley & Sons, 2005.

[27] L. Yao, B. Zhu, and M. Shoaran, “Fast and accurate decoding of finger movements from ecog through riemannian features and modern machine learning techniques,” *Journal of Neural Engineering*, vol. 19, no. 1, p. 016037, 2022.

[28] A. Cichocki, D. Mandic, L. De Lathauwer, G. Zhou, Q. Zhao, C. Caiafa, and H. A. Phan, “Tensor decompositions for signal processing applications: From two-way to multiway component analysis,” *IEEE signal processing magazine*, vol. 32, no. 2, pp. 145–163, 2015.

[29] T. G. Kolda and B. W. Bader, “Tensor decompositions and applications,” *SIAM review*, vol. 51, no. 3, pp. 455–500, 2009.

[30] L. De Lathauwer, “Decompositions of a higher-order tensor in block terms—part i: Lemmas for partitioned matrices,” *SIAM Journal on Matrix Analysis and Applications*, vol. 30, no. 3, pp. 1022–1032, 2008.

[31] ——, “Decompositions of a higher-order tensor in block terms—part ii: Definitions and uniqueness,” *SIAM Journal on Matrix Analysis and Applications*, vol. 30, no. 3, pp. 1033–1066, 2008.

[32] L. De Lathauwer and D. Nion, “Decompositions of a higher-order tensor in block terms—part iii: Alternating least squares algorithms,” *SIAM journal on Matrix Analysis and Applications*, vol. 30, no. 3, pp. 1067–1083, 2008.

[33] B. Hunyadi, D. Camps, L. Sorber, W. Van Paesschen, M. De Vos, S. Van Huffel, and L. De Lathauwer, “Block term decomposition for modelling epileptic seizures,” *EURASIP Journal on Advances in Signal Processing*, vol. 2014, no. 1, pp. 1–19, 2014.

[34] Q. Zhao, C. F. Caiafa, D. P. Mandic, Z. C. Chao, Y. Nagasaka, N. Fujii, L. Zhang, and A. Cichocki, “Higher order partial least squares (hopls): a generalized multilinear regression method,” *IEEE transactions on pattern analysis and machine intelligence*, vol. 35, no. 7, pp. 1660–1673, 2012.

[35] A. Eliseyev and T. Aksenova, “Penalized multi-way partial least squares for smooth trajectory decoding from electrocorticographic (ecog) recording,” *PloS one*, vol. 11, no. 5, p. e0154878, 2016.

[36] K. J. Miller and G. Schalk, “Prediction of finger flexion: 4th brain-computer interface data competition,” *BCI Competition IV*, vol. 1, pp. 1–2, 2008.

[37] L. R. Tucker, “Some mathematical notes on three-mode factor analysis,” *Psychometrika*, vol. 31, no. 3, pp. 279–311, 1966.

[38] L. De Lathauwer, B. De Moor, and J. Vandewalle, “A multilinear singular value decomposition,” *SIAM journal on Matrix Analysis and Applications*, vol. 21, no. 4, pp. 1253–1278, 2000.

[39] S. Zubair and W. Wang, “Tensor dictionary learning with sparse tucker decomposition,” in *2013 18th international conference on digital signal processing (DSP)*. IEEE, 2013, pp. 1–6.

[40] A. H. Phan and A. Cichocki, “Fast and efficient algorithms for nonnegative tucker decomposition,” in *International Symposium on Neural Networks*. Springer, 2008, pp. 772–782.

[41] L. De Lathauwer, B. De Moor, and J. Vandewalle, “On the best rank-1 and rank-(r 1, r 2,..., rn) approximation of higher-order tensors,” *SIAM journal on Matrix Analysis and Applications*, vol. 21, no. 4, pp. 1324–1342, 2000.

[42] M. Mørup and L. K. Hansen, “Sparse coding and automatic relevance determination for multi-way models,” in *SPARS’09-Signal Processing with Adaptive Sparse Structured Representations*, 2009.

[43] T. Yokota and A. Cichocki, “Multilinear tensor rank estimation via sparse tucker decomposition,” in *2014 Joint 7th International Conference on Soft Computing and Intelligent Systems (SCIS) and 15th International Symposium on Advanced Intelligent Systems (ISIS)*. IEEE, 2014, pp. 478–483.

[44] T. Yokota, N. Lee, and A. Cichocki, “Robust multilinear tensor rank estimation using higher order singular value decomposition and information criteria,” *IEEE Transactions on Signal Processing*, vol. 65, no. 5, pp. 1196–1206, 2016.

[45] Q. Shi, H. Lu, and Y.-m. Cheung, “Tensor rank estimation and completion via cp-based nuclear norm,” in *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, 2017, pp. 949–958.

[46] G. I. Allen and M. Maletić-Savatić, “Sparse non-negative generalized

pca with applications to metabolomics," *Bioinformatics*, vol. 27, no. 21, pp. 3029–3035, 2011.

[47] G. Schwarz *et al.*, "Estimating the dimension of a model," *Annals of statistics*, vol. 6, no. 2, pp. 461–464, 1978.

[48] C. M. Bishop, "Training with noise is equivalent to tikhonov regularization," *Neural computation*, vol. 7, no. 1, pp. 108–116, 1995.

[49] F. Camarrone and M. M. Van Hulle, "Fast algorithm for multiway regression," in *2017 22nd International Conference on Digital Signal Processing (DSP)*, 2017, pp. 1–5.

[50] C. D. Binnie, *Clinical neurophysiology: EEG, paediatric neurophysiology, special techniques and applications*. Elsevier Health Sciences, 2003, vol. 2.

[51] F. Wilcoxon, "Individual comparisons by ranking methods," in *Breakthroughs in statistics*. Springer, 1992, pp. 196–202.

Axel Faes received the M.Sc. degree in computer science engineering from the Katholieke Universiteit Leuven (KU Leuven), Leuven, Belgium. He also received the M.SC. degree in artificial intelligence from the same university. Since 2018, he is a PhD student of the Computational Neuroscience Group of the Laboratory for Neuro- and Psychophysiology at KU Leuven, Leuven, Belgium. His research interests include brain-computer interface, graphical modelling, signal processing and motor control behavior.

Flavio Camarrone received the M.Sc. degree in computer engineering from the Roma Tre University, Rome, Italy. In 2014, he became a PhD student of the Computational Neuroscience Group of the Laboratory for Neuro- and Psychophysiology at KU Leuven, Leuven, Belgium. In 2019, he obtained his PhD in Biomedical Sciences from the same university. His research interests include brain-computer interface, signal processing, machine learning, and motor control behavior.

Marc M. Van Hulle received the M.Sc. degree in electrotechnical engineering and the Ph.D. degree in applied sciences from the Katholieke Universiteit Leuven (KU Leuven), Leuven, Belgium. He also received the B.Sc.Econ. and M.B.A. degrees. He received the Doc- tor Technices degree from Queen Margrethe II of Denmark, in 2003, and an Honorary Doctoral degree from Brest State University, Brest, Belarus, in 2009. He is currently a Full Professor at the KU Leuven Medical School, where he heads the Computational Neuro-science Group of the Laboratorium voor Neuro- en Psychofysiologie. In 1992, he was with the Brain and Cognitive Sciences Department, Massachu- setts Institute of Technology, Boston, as a Post-Doctoral Scientist. He has authored a monograph titled Faithful Representa-tions and Topo- graphic Maps: From Distortion- to Information-Based Self-Organization (John Wiley, 2000; also translated into Japanese) and 250 technical publications. His current research interests include computational neuroscience, brain computer interfacing, neural networks, data mining, and signal processing. Prof. Van Hulle is a member of the Belgian Royal Academy of Medicine and the Spanish Royal National Acad- emy of Pharmacy, Institute of Spain.