Journal of Neural Engineering

RECEIVED
6 July 2022

REVISED
28 September 2022

ACCEPTED FOR PUBLICATION
14 October 2022

PUBLISHED
14 November 2022

PAPER

Finger movement and coactivation predicted from intracranial brain activity using extended block-term tensor regression

A Faes* and M M Van Hulle

Department of Neurosciences, Laboratory for Neuro- & Psychophysiology, KU Leuven—University of Leuven, B-3000 Leuven, Belgium
* Author to whom any correspondence should be addressed.

E-mail: axel.faes@kuleuven.be

Keywords: tensor decomposition, partial least squares, multiway data, tensor, electrocorticography, finger movement decoding, brain computer interfaces

Supplementary material for this article is available online

Abstract

Objective. We introduce extended Block-Term Tensor Regression (eBTTR), a novel regression method designed to account for the multilinear nature of human intracranial finger movement recordings. *Approach.* The proposed method relies on recursive Tucker decomposition combined with automatic component extraction. *Main results.* eBTTR outperforms state-of-the-art regression approaches, including multilinear and deep learning ones, in accurately predicting finger trajectories as well as unintentional finger coactivations. *Significance.* eBTTR rivals state-of-the-art approaches while being less computationally expensive which is an advantage when intracranial electrodes are implanted acutely, as part of the patient's presurgical workup, limiting time for decoder development and testing.

1. Introduction

Every year, up to half a million patients worldwide face paralysis due to spinal cord injury, brainstem stroke and amyotrophic lateral sclerosis (ALS) [1]. Brain computer interfaces (BCIs) are capable of bypassing disconnected neural pathways to replace the function of a lost or impaired body part, which led them to becoming promoted as a solution for these patients. Typically, BCI systems consist of several components: from the recorded brain activity, signal features are extracted and the result translated ('decoded') into commands controlling an external device such as a robotic arm or hand. Extraordinary results have been achieved with BCI controlling functional electrical stimulation of hand muscles [2, 3] and prosthetic hands and arms, exoskeletons or other effectors [4–7].

Motor BCI researchers are increasingly paying attention to electrocorticography (ECoG) where electrodes are placed directly above the cortex (i.e. sub- or epidurally) thereby avoiding tissue damage and histological processes wire microelectrode implants are prone to. ECoG signals yield significantly higher amplitudes compared to scalp EEG, are not contaminated by artifacts, enjoy a broader bandwidth and higher spatial resolution [8, 9], as well as long-term signal stability [10, 11].

Ramsey and co-workers at UMC Utrecht [12] reported success on the real-time, long-term use of an ECoG-based motor BCI in a patient with late-stage ALS. The patient imagined closing the right hand to select a cued target (i.e. a displayed character), while the decoder was detecting changes in the ECoG spectrum. Benabid *et al* [13] reported a major breakthrough with a tetraplegic patient controlling a four-limb exoskeleton with up to 8 degrees of freedom (start/stop walking, arm trajectory, arm- and wrist rotations). Despite this success, Benabid *et al* admitted that higher resolution ECoG electrode grids and further algorithmic developments are required to achieve better skilled movements of the joints and in particular of the hand (such as prehension and faster movements of the fingers). When looking beyond movement/no-movement classification, decoding finger movement trajectories from ECoGs remains an unsolved problem, even though a few attempts have been made. For example, Kubanek *et al* [14] were the first to continuously decode performed flexions/extensions of individual fingers by analyzing local motor potentials (LMP) and spectral amplitudes in five frequency bands. The authors used a conventional sparse linear regression model and obtained an average decoding performance of $r = 0.52$ (predicted vs. expected correlation). Since then, continuous finger movement decoding has been studied based on Gaussian [15], linear regression [16, 17], convolutional neural networks (CNNs), random forests (RFs) and deep learning networks (proposed in [18]). However, finger trajectory decoding could benefit from recent developments in multilinear algebra (multiway decoding) as ECoG is in essence structured in space, time, and frequency domains. This structure is largely ignored in traditional vector- or matrix-based regression models, as the data is concatenated into vectors and matrices (aka unfolding). Multiway models preserve the multilinear structure of the data and support the discovery of potentially hidden multilinear components [19]. The two most popular multiway frameworks are Tucker (TKD) and CANDECOMP/PARAFAC (or CPD) decompositions (see [19, 20] for review) and both aim to determine the low-dimensional space where important information is residing.

De Lathauwer introduced a tensor decomposition algorithm, called Block Term Decomposition (BTD) [21–23], which can be seen as a unified version of TKD and CPD. More specifically, in [23] two BTD models were proposed to approximate Nth-order tensor data as a sum of K-blocks called rank-$(L_1^k,\dots,L_N^k)$ BTD and rank-$(L_1,\dots,L_N)$ BTD with $k = 1,\dots,K$. In the former model, blocks can have different multilinear rank (MTR), where 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 $= (1,1,\dots,1)$, then it reduces to CPD. This approach provides great flexibility and opens new possibilities for multiway data analysis, above all when rank-$(L_1^k,\dots,L_N^k)$ BTD is adopted. For instance, an EEG study [24] showed that, while CPD failed to model epileptic seizures, rank-$(L_1,\dots,L_N)$ BTD correctly extracted the ictal sources that matched clinical assessment.

The abovementioned multiway approaches have been adapted for regression analysis to model the relationship between arm trajectory and ECoG signals recorded from monkeys [25, 26] and recently between exoskeleton-based arm trajectory, arm- and wrist rotations and ECoG signals recorded from the tetraplegic patient mentioned above [13]. 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 [25]. On the other hand, Zhao *et al* [25] developed a powerful generalized framework, called Higher-Order Partial Least Squares (HOPLS), based on $(1, L_2,\dots,L_N)$-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.

Camarrone and co-workers recently introduced Block-Term Tensor Regression (BTTR) [27, 28] and showed it performed with similar accuracy as HOPLS while being much faster to train. However, unlike BTTR, HOPLS generalises to predict a tensor (multiway array) $\underline{\mathbf{Y}}$ from a tensor $\underline{\mathbf{X}}$ by projecting the data onto latent space and performing regression on the corresponding latent variables. We introduce a similar generalization but for Block-Term Tensor Regression, further called eBTTR, while retaining its computational advantage over HOPLS and apply it to accurately decode finger movements, even unintended ones due to limited finger independence, further referred to as finger coactivation.

2. Materials and methods

One of the main limitations of BTTR is that it can predict a scalar variable only. This becomes an issue when modeling coordinated finger movements as it in the case of grasping a cup. One evident solution is to have a BTTR model per finger. However, this limits exploiting information shared between fingers. More fundamentally, unlike individual fingers being represented at separate anatomical locations [29], when flexing multiple fingers simultaneously, the same locations now exhibit spatially sparse and even mixed signals [30]. This implies that decoders trained on individual finger movements could become inadequate when representing coordinated finger movements.

To tackle this, we extend BTTR into eBTTR to model multiple variables simultaneously. As an example case we consider intended single finger movement in the presence of unintended coactivation of the other fingers. An overview of the mathematical notation used can be found in table 1.

Conceptually, the (e)BTTR algorithm consists of the following operations. First, the data is cast into tensor format, $\underline{\mathbf{X}}$, with the corresponding 'labels' into matrix format, $\mathbf{Y}$. (e)BTTR proceeds iteratively, yielding a series of blocks with declining contribution to the regression performance: at each iteration, a mode-1 cross-covariance tensor $\underline{\mathbf{C}}$ is constructed by combining information from both $\underline{\mathbf{X}}$ and $\mathbf{Y}$. This cross-covariance tensor $\underline{\mathbf{C}}$ is then decomposed and ACE (Automatic Parameter Extraction) used for automatic parameter estimation of the blocks. The resulting decomposition can be used to reconstruct $\underline{\mathbf{X}}$ and $\mathbf{Y}$ and thus comprise the learned regression model (from this iteration). $\underline{\mathbf{X}}$ and $\mathbf{Y}$ get deflated before the

Table 1. 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
$\lvert \cdot_F \rvert$	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*
$[\![\underline{\mathbf{C}}; \mathbf{M}^{(1)}, \ldots, \mathbf{M}^{(N)}]\!]$	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

next iteration starts. After training, the learned model can be used to predict $\mathbf{Y}$ using a $\underline{\mathbf{X}}_{\text{test}}$.

Similar to HOPLS, BTTR and its multivariate extension eBTTR rely on a partial least squares approach (PLS). Due to this, their performance is similar to that of HOPLS. Conceptually, the main differences with HOPLS are: the decomposition of the mode-1 cross-product between predictor- and response variables, the blocks that can have different multilinear ranks (MTRs), and the use of the automatic component extraction (ACE) for parameter estimation. These differences cause (e)BTTR to be faster to train than HOPLS, with comparable accuracies.

2.1. Extended block-term tensor regression (eBTTR)

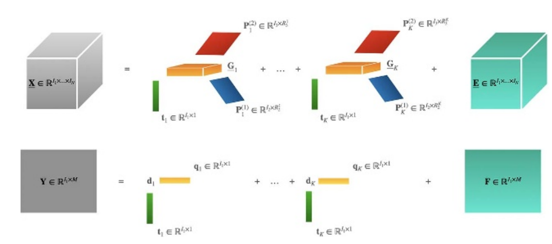
The proposed extended Block-Term Regression (eBTTR) model is based on $(L_1^k, \ldots, L_N^k)$ BTD with automatic MTR determination. Specifically, it is a deflation-based method that sequentially decomposes $\underline{\mathbf{X}}$ and $\mathbf{Y}$ into a series of blocks of maximally correlated representations extracted via ACE. A scheme of eBTTR is shown in figure 1.

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

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

$$\mathbf{Y}_{\text{train}} = \sum_{k=1}^{K} \mathbf{u}_k \mathbf{q}_k^T + \mathbf{F}_k \text{with } \mathbf{u}_k = \mathbf{t}_k b_k$$

with $\underline{\mathbf{G}}_k \in \mathbb{R}^{1 \times R_2^k \times \cdots \times R_N^k}$ the core tensor for the kth-block, $\mathbf{P}_k^{(n)}$ the kth loading matrix for the n-mode, $\mathbf{u}_k$ and $\mathbf{t}_k$ latent components, $\mathbf{q}_k$ the loading matrix, b_k the regression coefficient, and $\underline{\mathbf{E}}_k$ and $\mathbf{F}_k$ 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 follows: $\mathbf{Y}_{\text{test}} = \mathbf{TZ} = \mathbf{X}_{\text{test}(1)} \mathbf{WZ}$ where each column $\mathbf{w}_k = (\mathbf{P}_k^{(n)} \otimes \cdots \otimes \mathbf{P}_k^{(2)}) vec(\underline{\mathbf{G}}_k)$ and each row $z_k = b_k q_k$. This is summarized in figure 2.



$\underline{\mathbf{X}} \in \mathbb{R}^{I_1 \times \ldots \times I_N}$
=
$\mathbf{P}_1^{(2)} \in \mathbb{R}^{I_3 \times R_3^1}$
$\underline{\mathbf{G}}_1$
$\mathbf{t}_1 \in \mathbb{R}^{I_1 \times 1}$
$\mathbf{P}_1^{(1)} \in \mathbb{R}^{I_2 \times R_2^1}$
+ ... +
$\mathbf{P}_K^{(2)} \in \mathbb{R}^{I_3 \times R_3^K}$
$\underline{\mathbf{G}}_K$
$\mathbf{t}_K \in \mathbb{R}^{I_1 \times 1}$
$\mathbf{P}_K^{(1)} \in \mathbb{R}^{I_2 \times R_2^K}$
+
$\underline{\mathbf{E}} \in \mathbb{R}^{I_1 \times \ldots \times I_N}$
$\mathbf{Y} \in \mathbb{R}^{I_1 \times M}$
=
$\mathbf{d}_1$
$\mathbf{q}_1 \in \mathbb{R}^{I_1 \times 1}$
$\mathbf{t}_1 \in \mathbb{R}^{I_1 \times 1}$
+ ... +
$\mathbf{d}_K$
$\mathbf{q}_K \in \mathbb{R}^{I_1 \times 1}$
$\mathbf{t}_K \in \mathbb{R}^{I_1 \times 1}$
+
$\mathbf{F} \in \mathbb{R}^{I_1 \times M}$

Figure 1. Scheme of eBTTR algorithm with 3rd-order predictor variable $\underline{\mathbf{X}}$ and 2nd-order response variable Y. Note that each block is computed using ACE. In this work, $\underline{\mathbf{X}} \in \mathbb{R}^{\text{Samples} \times \text{Channels} \times \text{Frequencies} \times \text{Time}}$ and $\mathbf{Y} \in \mathbb{R}^{\text{Samples} \times \text{Fingers}}$.

```
Input: $\underline{\mathbf{X}} \in \mathbb{R}^{I_1 \times \ldots \times I_N}, \mathbf{Y} \in \mathbb{R}^{I_1 \times M}, K$
Output: $\{\mathbf{P}_k^{(n)}\}, \{\mathbf{t}_k\}, \{\mathbf{q}_k\}, \underline{\mathbf{G}}_k^{(X)}$ for $k = 1, \ldots, K$ ; $n = 2, \ldots, N$
 1: Initialisation of $\underline{\mathbf{E}}_1 = \underline{\mathbf{X}}$ and $\mathbf{F}_1 = \mathbf{Y}$
 2: for $k = 1$ to $K$ do
 3:   if $\|\underline{\mathbf{E}}_k\| > \epsilon$ and $\|\mathbf{F}_k\| > \epsilon$ then
 4:     $\underline{\mathbf{C}}_k = \langle \underline{\mathbf{E}}_k, \mathbf{F}_k \rangle_{\{1,1\}}$
 5:     $\underline{\mathbf{G}}_k^{(X)}, \mathbf{q}_k, \mathbf{t}_k, \mathbf{P}_k^{(2)}, \ldots, \mathbf{P}_k^{(N)} = ACE(\underline{\mathbf{E}}_k, \mathbf{F}_k)$
 6:     $\mathbf{u}_k = \mathbf{F}_k \mathbf{q}_k$
 7:     $\mathbf{d}_k = \mathbf{u}_k^T \mathbf{t}_k$
        {Deflation:}
 8:     $\underline{\mathbf{E}}_{k+1} = \underline{\mathbf{E}}_k - [\![\underline{\mathbf{G}}_k^{(X)}; \mathbf{t}_k, \mathbf{P}_k^{(2)}, \ldots, \mathbf{P}_k^{(N)}]\!]$
 9:     $\mathbf{F}_{k+1} = \mathbf{F}_k - \mathbf{d}_k \mathbf{t}_k \mathbf{q}_k^T$
10:   else
11:     break
12:   end if
13: end for
```

Figure 2. eBTTR.

2.1.1. Automatic component extraction (ACE)

Given an N-way variable $\underline{\mathbf{X}} \in \mathbb{R}^{I_1 \times \cdots \times I_N}$ and a vectorial variable $\mathbf{Y} \in \mathbb{R}^{I_1 \times M}$, we aim to automatically extract the latent components $\mathbf{t}, \mathbf{q}$ and $P^{(n)}{}_{(n=2)}^{N}$, associated with the nth mode of $\underline{\mathbf{X}}$ and maximally correlated with $\mathbf{Y}$, while $|\underline{\mathbf{X}} - [\![\underline{\mathbf{G}}; \mathbf{t}, \mathbf{P}^{(2)}, \ldots, \mathbf{P}^{(N)}]\!]|_F$ is minimized.

Within ACE, we define the mode-1 cross-product between predictor and response variables as $\underline{\mathbf{C}} = \langle \underline{\mathbf{X}}, \mathbf{Y} \rangle_{(1)}$ and its decomposition as $\underline{\mathbf{C}} \approx [\![\underline{\mathbf{G}}^{(c)}; \mathbf{q}, \mathbf{P}^{(2)}, \ldots, \mathbf{P}^{(N)}]\!]$. We provide the model with automatic SNR and τ selection based on Bayesian Information Criterion (BIC) defined here as:

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

```
Input: $\underline{\mathbf{X}} \in \mathbb{R}^{I_1\times\ldots\times I_N}$, $\mathbf{Y} \in \mathbb{R}^{I_1\times M}$
Output: $\underline{\mathbf{G}}^{(X)} \in \mathbb{R}^{1\times R_2\times\ldots\times R_N}$, $\mathbf{q}, \mathbf{t}, \{\mathbf{P}^{(n)}\}_{n=2}^N$
 1: $\underline{\mathbf{C}} = \langle \underline{\mathbf{X}}, \mathbf{Y}\rangle_{(1)}$
 2: Initialisation of $\tau = 90, ..., 100$; $SNR = 1, ..., 50$
 3: for $SNR_i$ in $SNR$ do
 4:     for $\tau_j$ in $\tau$ do
 5:         $\underline{\mathbf{G}}, \mathbf{q}, \{\mathbf{P}^{(n)}\}_{n=2}^N = mPSTD(\underline{\mathbf{X}}, \mathbf{Y}, SNR_i, \tau_j)$
 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: $\underline{\mathbf{G}}, \mathbf{q}, \{\mathbf{P}^{(n)}\}_{n=2}^N = mPSTD(\underline{\mathbf{X}}, \mathbf{Y}, SNR^*, \tau^*)$
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: $\underline{\mathbf{G}}^{(X)} = [\![\underline{\mathbf{X}}; \mathbf{t}^T, \mathbf{P}^{(2)T}, ..., \mathbf{P}^{(N)T}]\!]$
16: return $\underline{\mathbf{G}}^{(X)}, \mathbf{q}, \mathbf{t}, \{\mathbf{P}^{(n)}\}_{n=2}^N$
```

Figure 3. ACE.

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

Figure 4. mPSTD.

where $\underline{\mathbf{G}}^{(c)}$, q and $\{\mathbf{P}^{(n)}\}_{n=2}^N$ are the sparse core, latent vector and factor matrices obtained with mPSTD [28]—a modified version of the original sparse Tucker decomposition (PSTD) [31]—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 [32]. 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 $\underline{\mathbf{G}}^{(c)}$, q 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. This is summarized in figure 3.

The mPSTD model is first initialized with higher-order orthogonal iteration (HOOI) [33]. Then, iteratively, a soft-thresholding rule based on parameter λ, alternated with a threshold τ, are applied to enhance model sparsity and to prune irrelevant components, respectively. Note that in [31] SNR $\in [1, 50]$ is used to derive, via a line search, the optimal degree of sparsity λ of the core tensor (see [31]. At each iteration, the core tensor $\underline{\mathbf{G}}$ is updated using the soft-thresholding rule as $\underline{\mathbf{G}} = \mathrm{sgn}(\underline{\mathbf{G}}) \times \max\{|\underline{\mathbf{G}}| - \lambda, 0\}$, while the threshold $\tau \in [0, 100]$ is used to reject unnecessary components from the n-mode $S^{(n)} = \left\{ r | 100 \left(1 - \frac{\sum_i \mathbf{G}_{(n)(r,i)}}{\sum_{t,i} \mathbf{G}_{(n)(t,i)}}\right) \geqslant \tau \right\}$, $\mathbf{P}^{(n)} = \mathbf{P}^{(n)}(:, S^{(n)})$, $\mathbf{q} = \mathbf{q}(S^{(n)})$ and $\mathbf{G}^{(n)} = \mathbf{G}^{(n)}(S^{(n)}, :)$. The mPSTD is summarized in figure 4.

2.2. Non-multilinear approaches

We will consider the BCI competition IV dataset (see further) for comparing the performance of eBTTR with the winner of this competition, a linear regression model based on amplitude modulation (AM) [17], as well as with more recent attempts based on RFs, CNNs, and Long short-term memory network (LSTM) (proposed and compared in [18]), all of which are non-multilinear approaches.

Since in both [17, 18] 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 [17]. First, we filtered the ECoG signals in the sub-gamma (1–60 Hz), 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 [17].

For LARS, we used the *LassoLars* function of 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). 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 of 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 [18]. 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).

2.3. Dataset

We will compare eBTTR's performance with that of the aforementioned non-multilinear models, as well as with multilinear models HOPLS and BTTR, in predicting continuous finger flexions from ECoG recordings. We will use hereto the publically-available BCI Competition IV dataset 4 for reproducibility's sake. It comprises ECoG signals sampled at 1000 Hz from the motor cortex (hand-knob area) of three subjects, as well as the time courses of the flexion of each of five fingers of the contralateral hand (gauged with a data glove). There are 150 trials (samples) in total (30 trials per finger) recorded in a single session (600 s). Subjects flexed the cued finger 3–5 times for 2 s followed by a rest period of 2 s. The first 400 and the last 200 s of recording were used as training and testing sets, respectively. The number of ECoG electrodes (channels) was 62 for Subject 1, 48 for Subject 2, and 64 for Subject 3. More details about the data can be found in [34].

The dataglove position measurement lags by 37 ms ($\pm$3 ms, SEM) the amplifier measurement. However, this is of the same order of granularity as the datagloves position measurement as it is sampled at 25 Hz, thus every 40 ms. Hence, the dataglove position measurement is shifted by 1 position in order to account for the lag. This is also done in the approaches against which we compare our (e)BTTR's performance.

The glove data and the ECoG signals were extracted starting 1 s prior to trial onset ('epoch'). ECoG recordings were prepared first by filtering out the power line using notch filters centered at 50 and 100 Hz, then, by removing bad channels (i.e. those exhibiting unstable or unchanging signals, i.e. channels 55 in subject 1, channels 21 and 38 in subject 2, and channel 50 in subject 3). The remaining channels were re-referenced to a Common Average Reference (CAR) [3]: the average of all signals is taken as reference and subtracted from all signals [35]. The ECoG signals are transformed into 4th order ECoG tensor $\underline{\mathbf{X}} \in \mathbb{R}^{\text{Samples}\times\text{Channels}\times\text{Frequencies}\times\text{Time}}$ as follows:

- **Samples** depends on the length of the data glove's trajectory vector Y.
- **Channels** corresponds to the number of curated electrodes (thus, after removing bad ones), i.e. 61 for subject 1, 46 for subject 2, and 63 for subject 3.
- **Frequencies** corresponds to the 8 bidirectional fourth-order Butterworth band-pass filters [36] ECoG signals are subjected to extract the corresponding spectral amplitudes in the 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) bands.
- **Time** is composed of 10 instances or bins as for each of the described 8 components, the most recent 1 s epoch is downsampled to 10 Hz.

The data glove data is normalized (z-scored) independently for each finger, yielding a vector $Y \in \mathbb{R}^{\text{Samples}\times\text{Fingers}}$. **Samples** corresponds to the sampled finger flexions over time. The **Fingers** dimension corresponds to the five fingers, Thumb, Index, Middle, Ring and Pinky.

As a result, for each channel c, a 3rd order ECoG tensor $\underline{\mathbf{X}}_c \in \mathbb{R}^{\text{Samples}\times\text{Frequencies}\times\text{Time}}$ is computed after epoch selection, band-pass filtering and 10 Hz downsampling. These steps are repeated for each time sample s. The results are then merged into a matrix $\mathbf{X}_{c,f} \in \mathbb{R}^{\text{Samples}\times\text{Time (10 bins)}}$. This matrix is then normalized (z-scored) to reduce the difference in magnitude between frequency bands: $x_{s,t} = (x_{s,t} - \mu_{c,f})/\delta_{c,f}$ where $\mu c,f$ and $\delta c,f$ are, respectively, the mean and standard deviation computed for channel c and frequency band f of the training set. Note that the same values are used to normalize the test set.

Once all $\mathbf{X}_{c,f} \in \mathbb{R}^{\text{Samples}\times\text{Time (10)}}$ are computed for the various components (8), they are merged into $\underline{\mathbf{X}}_c \in \mathbb{R}^{\text{Samples}\times\text{Frequencies (8)}\times\text{Time (10)}}$.

Next, when all 3rd order ECoG tensors $\underline{\mathbf{X}}_c \in \mathbb{R}^{\text{Samples}\times\text{Frequencies (8 components)}\times\text{Time (10 bins)}}$ are computed for each channel c, they are merged into $\underline{\mathbf{X}} \in \mathbb{R}^{\text{Samples}\times\text{Channels }\{61,\,46,\,63\}\times\text{Frequencies (8)}\times\text{Time (10)}}$.

2.4. Parameter optimization multilinear models, performance assessment

In order to optimize the model parameters from the training data, $K, R_2, \ldots, R_N$ for HOPLS and K for BTTR and eBTTR, a five-fold cross-validation approach was used.

Since the cited BCI Competition IV dataset 4 studies reported Pearson's correlation coefficients between data glove- and predicted intended finger trajectories, we also applied it here. In support of the statistical analysis, the test data was split in five non-overlapping blocks. We used the two-tailed Wilcoxon signed-rank test [37] to compare average accuracies per finger and subject; they are considered significantly different if the p-value is <0.05. The non-intended finger movements (coactivations) are visually inspected from hand avatar movies.

3. Results

The Pearson correlation coefficients are reported for the intended movements of each finger individually, and averaged across all fingers except for finger 4

Table 2. Intended (cued) finger movement accuracy for subject 1 (Pearson correlation, with significant differences to eBTTR in bold).

Methods	Thumb	Index	Middle	Ring	Pinky	Avg.
eBTTR	0.71 ± .04	0.75 ± .06	0.49 ± .06	0.69 ± .04	0.68 ± .01	0.66 ± .03
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**

Table 3. Idem to table 2 but for subject 2.

Methods	Thumb	Index	Middle	Ring	Pinky	Avg.
eBTTR	0.63 ± .05	0.47 ± .08	0.33 ± .03	0.52 ± .02	0.47 ± .01	0.48 ± .05
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**

Table 4. Idem to table 2 but for subject 3.

Methods	Thumb	Index	Middle	Ring	Pinky	Avg.
eBTTR	0.71 ± .05	0.57 ± .07	0.64 ± .04	0.62 ± .02	0.73 ± .01	0.66 ± .05
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**

(ring) as flexing the latter is difficult to suppress when the 3rd or 5th finger is flexing, as this was done by the authors of the cited papers (note that we assess coactivations qualitatively, see further). The average results and their standard deviations for the five blocks of test data (see above) are listed in tables 2–4 (listed under Avg.) for subjects 1, 2 and 3, respectively.

A statistically significant difference between BTTR and eBTTR was found for the middle finger of subject 1 and subject 2 and between eBTTR and HOPLS for subjects 1 and 2, mainly due to the difference in correlation coefficients of the middle finger.

We also investigated the runtime needed to estimate the model parameters. One of the main advantages of BTTR is its speed. BTTR is trained significantly faster than HOPLS (e.g. 3 min against 14 h for HOPLS). 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). Since eBTTR is based on BTTR, it enjoys the latter's fast training time (e.g. 5 min against 3 min for BTTR).

Figure 5 shows stills of the hand avatar replicating the predicted movements of a few models. Compared to BTTR and AM, we observe that eBTTR and HOPLS better decode joint finger movements, comprising intentional- (cued finger) and unintentional ones (coactivations)[1]. The latter are in part due to the subject's hand- and forearm tendon anatomy and the divergent connections from hand cortical motor neurons to spinal motor neurons, limiting cortical access to individual fingers [38]. Co-activations are therefore present in the data glove data, but most likely not encoded by the ECoG signals. One could question whether these coactivations should be decoded at all. However, rendering them makes the outcome look more natural and

[1] For a quantitative comparison in the case of eBTTR and BTTR, we refer to the supplementary material section.

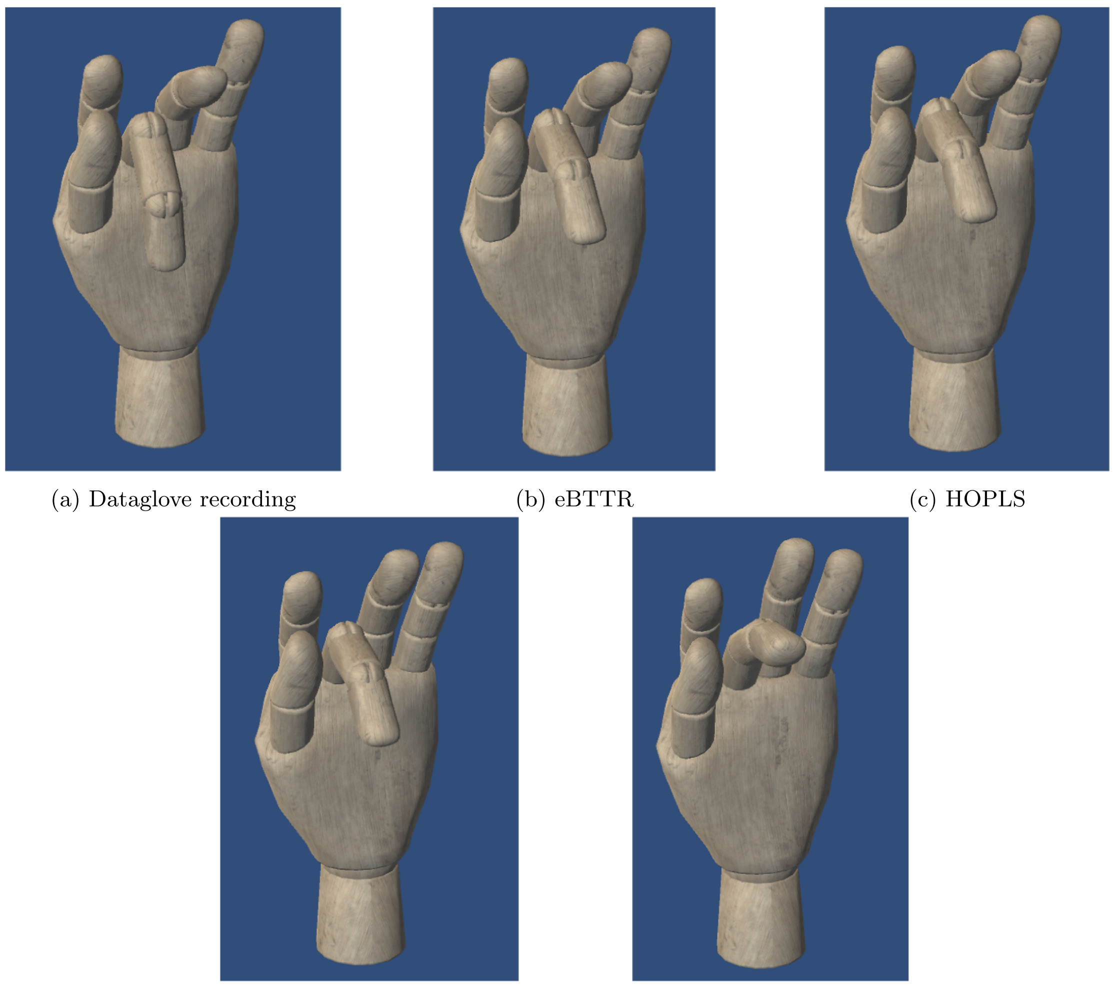

(a) Dataglove recording (b) eBTTR (c) HOPLS (d) BTTR (e) Non-multilinear approach: amplitude modulation (AM)

Figure 5. Unintentional coactivation for subject 3 during middle finger movement.

could provide additional sensory information when approaching a physical object.

4. Conclusion

Tensor-based decoders are able to exploit multiway structured data better than those that rely on data unfolding. Decoding joint finger movements is a challenging task that needs to be addressed when envisaging BCI-based hand prosthetic- or exoskeleton control in realistic object handling scenarios. To tackle this, we proposed an extension of the tensor decomposition approach we developed before so that it can be used to regress over multiple fingers simultaneously and showed that it can challenge state-of-the-art multiway regression techniques while outperforming more conventional ones.

Data availability statement

All data that support the findings of this study are included within the article (and any supplementary files).

Funding

A F was supported by the Belgian Fund for Scientific Research—Flanders (FWO 1157019N). M M V H 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 (G0A4118N, G0A4321N, G0C1522N), and the Hercules Foundation (AKUL 043).

ORCID iD

A Faes https://orcid.org/0000-0002-1637-255X

References

[1] WHO spinal cord injury (available at: www.who.int/news-room/fact-sheets/detail/spinal-cord-injury) (Accessed 30 June 2022)

[2] Bolu Ajiboye A *et al* 2017 Restoration of reaching and grasping movements through brain-controlled muscle stimulation in a person with tetraplegia: a proof-of-concept demonstration *Lancet* **389** 1821–30

[3] Bouton C E *et al* 2016 Restoring cortical control of functional movement in a human with quadriplegia *Nature* **533** 247–50

[4] Hochberg L R, Serruya M D, Friehs G M, Mukand J A, Saleh M, Caplan A H, Branner A, Chen D, Penn R D and Donoghue J P 2006 Neuronal ensemble control of prosthetic devices by a human with tetraplegia *Nature* **442** 164–71

[5] Collinger J L *et al* 2013 High-performance neuroprosthetic control by an individual with tetraplegia *Lancet* **381** 557–64

[6] Hochberg L R *et al* 2012 Reach and grasp by people with tetraplegia using a neurally controlled robotic arm *Nature* **485** 372–5

[7] Wodlinger B, Downey J E, Tyler-Kabara E C, Schwartz A B, Boninger M L and Collinger J L 2014 Ten-dimensional anthropomorphic arm control in a human brain–machine interface: difficulties, solutions and limitations *J. Neural Eng.* **12** 016011

[8] Miller K J, Sorensen L B, Ojemann J G, Den Nijs M 2009 Power-law scaling in the brain surface electric potential *PLoS Comput. Biol.* **5** e1000609

[9] Staba R J, Wilson C L, Bragin A, Fried I and Engel Jr J 2002 Quantitative analysis of high-frequency oscillations (80–500 Hz) recorded in human epileptic hippocampus and entorhinal cortex *J. Neurophysiol.* **88** 1743–52

[10] Wang W *et al* 2013 An electrocorticographic brain interface in an individual with tetraplegia *PLoS One* **8** e55344

[11] Nurse E S, John S E, Freestone D R, Oxley T J, Ung H, Berkovic S F, O'Brien T J, Cook M J and Grayden D B 2017 Consistency of long-term subdural electrocorticography in humans *IEEE Trans. Biomed. Eng.* **65** 344–52

[12] Vansteensel M J *et al* 2016 Fully implanted brain–computer interface in a locked-in patient with als *New Engl. J. Med.* **375** 2060–6

[13] Benabid A L *et al* 2019 An exoskeleton controlled by an epidural wireless brain–machine interface in a tetraplegic patient: a proof-of-concept demonstration *Lancet Neurol.* **18** 1112–22

[14] Kubanek J, Miller K J, Ojemann J G, Wolpaw J R and Schalk G 2009 Decoding flexion of individual fingers using electrocorticographic signals in humans *J. Neural Eng.* **6** 066001

[15] Wang Z, Ji Q, Miller K J and Schalk G 2010 Decoding finger flexion from electrocorticographic signals using a sparse Gaussian process *2010 20th Int. Conf. on Pattern Recognition* (IEEE) pp 3756–9

[16] Flamary R and Rakotomamonjy A 2012 Decoding finger movements from ECoG signals using switching linear models *Front. Neurosci.* **6** 29

[17] Liang N and Bougrain L 2012 Decoding finger flexion from band-specific ECoG signals in humans *Front. Neurosci.* **6** 91

[18] Xie Z, Schwartz O and Prasad A 2018 Decoding of finger trajectory from ECoG using deep learning *J. Neural Eng.* **15** 036009

[19] Cichocki A, Mandic D, De Lathauwer L, Zhou G, Zhao Q, Caiafa C and Phan H A 2015 Tensor decompositions for signal processing applications: from two-way to multiway component analysis *IEEE Signal Process. Mag.* **32** 145–63

[20] Kolda T G and Bader B W 2009 Tensor decompositions and applications *SIAM Rev.* **51** 455–500

[21] De Lathauwer L 2008 Decompositions of a higher-order tensor in block terms–part I: lemmas for partitioned matrices *SIAM J. Matrix Anal. Appl.* **30** 1022–32

[22] De Lathauwer L 2008 Decompositions of a higher-order tensor in block terms–part II: definitions and uniqueness *SIAM J. Matrix Anal. Appl.* **30** 1033–66

[23] De Lathauwer L and Nion D 2008 Decompositions of a higher-order tensor in block terms–part III: alternating least squares algorithms *SIAM J. Matrix Anal. Appl.* **30** 1067–83

[24] Hunyadi Bala, Camps D, Sorber L, Paesschen W V, De Vos M and De Lathauwer L 2014 Block term decomposition for modelling epileptic seizures *EURASIP J. Adv. Signal Process.* **2014** 1–19

[25] Zhao Q, Caiafa C F, Mandic D P, Chao Z C, Nagasaka Y, Fujii N, Zhang L and Cichocki A 2012 Higher order partial least squares (HOPLS): a generalized multilinear regression method *IEEE Trans. Pattern Anal. Mach. Intell.* **35** 1660–73

[26] Eliseyev A and Aksenova T 2016 Penalized multi-way partial least squares for smooth trajectory decoding from electrocorticographic (ECoG) recording *PLoS One* **11** e0154878

[27] Camarrone F, Branco M P, Ramsey N F and Van Hulle M M 2020 Accurate offline asynchronous detection of individual finger movement from intracranial brain signals using a novel multiway approach *IEEE Trans. Biomed. Eng.* **68** 2176–87

[28] Faes A, Camarrone F and Van Hulle M M 2022 Single finger trajectory prediction from intracranial brain activity using block-term tensor regression with fast and automatic component extraction *IEEE Trans. Neural Netw. Learn. Syst.* accepted (https://doi.org/10.1109/TNNLS.2022.3216589)

[29] Miller K J, Honey C J, Hermes D, Rao R P, denNijs M and Ojemann J G 2014 Broadband changes in the cortical surface potential track activation of functionally diverse neuronal populations *NeuroImage* **85** 711–20

[30] Pan G, Li J-J, Qi Y, Yu H, Zhu J-M, Zheng X-X, Wang Y-M and Zhang S-M 2018 Rapid decoding of hand gestures in electrocorticography using recurrent neural networks *Front. Neurosci.* **12** 555

[31] Yokota T and Cichocki A 2014 Multilinear tensor rank estimation via sparse tucker decomposition *2014 Joint 7th Int. Conf. on Soft Computing and Intelligent Systems (SCIS) and 15th Int. Symp. on Advanced Intelligent Systems (ISIS)* (IEEE) pp 478–83

[32] Allen G I and Maletić-Savatić M 2011 Sparse non-negative generalized pca with applications to metabolomics *Bioinformatics* **27** 3029–35

[33] De Lathauwer L, De Moor B and Vandewalle J 2000 On the best rank-1 and rank-$(R_1, R_2, \ldots, R_N)$ approximation of higher-order tensors *SIAM J. Matrix Anal. Appl.* **21** 1324–42

[34] Miller K J and Schalk G 2008 Prediction of finger flexion: 4th brain-computer interface data competition *BCI Competition IV* **1** 1–2

[35] Binnie C D 2003 *Clinical Neurophysiology: EEG, Paediatric Neurophysiology, Special Techniques and Applications* vol 2 (New York: Elsevier Health Sciences)

[36] Tucker L R 1966 Some mathematical notes on three-mode factor analysis *Psychometrika* **31** 279–311

[37] Wilcoxon F 1992 Individual comparisons by ranking methods *Breakthroughs in Statistics* (Berlin: Springer) pp 196–202

[38] Lang C E and Schieber M H 2004 Human finger independence: limitations due to passive mechanical coupling versus active neuromuscular control *J. Neurophysiol.* **92** 2802–10