International Journal of Electrical and Computer Engineering (IJECE)
Vol. 10, No. 5, October 2020, pp. 4721~4737
ISSN: 2088-8708, DOI: 10.11591/ijece.v10i5.pp4721-4737

Tracking times in temporal patterns embodied in intra-cortical data for controlling neural prosthesis an animal simulation study

Abdessalam Kifouche[1], Abderrezak Guessoum[2]
[1]University of Ghardaia, Algeria
[1,2]LATSI, University of Blida, Algeria

Article Info

Article history:

Received Sep 14, 2019
Revised Mar 14, 2020
Accepted Mar 25, 2020

Keywords:

Multiple input single output
Decoding algorithm
Spatiotemporal pattern
Spike
Time delay neural network

ABSTRACT

Brain-machines capture brain signals in order to restore communication and movement to disabled people who suffer from brain palsy or motor disorders. In brain regions, the ensemble firing of populations of neurons represents spatio-temporal patterns that are transformed into outgoing spatio-temporal patterns which encode complex cognitive task. This transformation is dynamic, non-stationary (time-varying) and highly nonlinear. Hence, modeling such complex biological patterns requires specific model structures to uncover the underlying physiological mechanisms and their influences on system behavior. In this study, a recent multi-electrode technology allows the record of the simultaneous neuron activities in behaving animals. Intra-cortical data are processed according to these steps: spike detection and sorting, than desired action extraction from the rate of the obtained signal. We focus on the following important questions about (i) the possibility of linking the brain signal time events with some time-delayed mapping tools; (ii) the use of some suitable inputs than others for the decoder; (iii) a consideration of separated data or a special representation founded on multi-dimensional statistics. This paper concentrates mostly on the analysis of parallel spike train when certain critical hypotheses are ignored by the data for the working method. We have made efforts to define explicitly whether the underlying hypotheses are actually achieved. In this paper, we propose an algorithm to define the embedded memory order of NARX recurrent neural networks to the hand trajectory tracking process. We also demonstrate that this algorithm can improve performance on inference tasks.

Corresponding Author:

Abdessalam Kifouche,
University of Ghardaia, Algeria.
Email: askifouche@gmail.com

1. INTRODUCTION

In this work, we aim to design a new hand trajectory decoder from brain signals in order to control a motor neuroprosthesis or at least know a desired motor action. Brain computer interface described in this paper is known in literature as a system to restore the lost motor functions because of a disease or an injury by activating paralyzed muscles. A number of previous studies have proved that desired actions can be extracted from motor cortex and, in turn, can be used as a reliable enough command signal to control a computer interface to realize actions with a robotic member for instance [1, 2]. Many technical ameliorations in microelectronics have been brought to help patients e.g. for epilepsy suppression [3], spinal cord injuries [4, 5], cochlear implant [6]. Research groups; in several papers; have previously confirmed that monkeys and human are able to learn how to control a robot limb or move a wheelchair [7, 8], or more simply, maintaining a communication with the external environment, easily by stimulating neuron groups that participate in normal arm movement [9, 10]. Technical stability, limited power source, flexibility in

critical system parameters, restricted space to host the system and real time control are essential for these prosthesis to become clinically viable. Therefore neuroprosthetic strategies are developing toward closed-loop command systems consisting 3 functional blocks: neural signal recording, processing and neuro-muscular stimulation as depicted in Figure 1.

To obtain and comprehend the signal processed in the biological network of neurons; spikes should be assigned to specific neural sources. This classification procedure is termed spike sorting. A proper sorting of spikes with respect to their origin ameliorate considerably the decoding operation. Then, before representing the position of the animal in terms of firing patterns, sorting spikes must be accomplished and the spike's rate of each neuron is calculated [11, 12]. Nevertheless, signal instabilities could arise from physical factors like the implants movement, reaction of the tissue or material degradation. Whatever the causes, signal changes can be significant, reaching 60% of the waveforms as reported by Dickey et al. [13]. In prior studies, we had used a multi-scale seriation approach for clustering spike trains [14].

The decoding function that links neural activity to behavior may be relatively unstable as well as degrading decoding performance. In [15], authors present right and left hand movements decoding by a probabilistic neural network using EEG signal and wavelet transform based on for classification and feature extraction. Perge et al. evaluated in [16, 17] the nature and extent of instability in spiking populations recorded in the context of an ongoing pilot clinical trial of people with tetraplegia: they found that systematic rate changes occur commonly and they can cause estimation errors in the decoded kinematic parameters leading to degraded performance that presents itself as a directional bias. Adaptive filters attempting to mitigate these instabilities exist but their efficacy has not been established yet [18].

Decoding brain activity has been extensively studied in the literature giving birth to various methods, see e.g. [9, 19]. A state space representation is adequate for this problem. In [20], Wu et al. modeled for example with a kalman filter (KF) the hand coordinates (acceleration, velocity and position on the two axes) with a probabilistic association between these motions and the brain signal. Kostov et al. [21] improved the possibility of real time control with a presence of a partial spinal cord injury by using an adaptive logic network. In [22], Gage et al. realized a control of cortical tasks by training a rat using a co-adaptive KF. These methods can estimate hidden states even in the absence of an accurate model of the system. In [23], Brockwell applied particle filter in order to get an estimation of the hidden states from cortical signals using the algorithm of Monte Carlo algorithm, with no hypothesis on the observations distribution. In summary, state space method offer a coherent outline for modeling stochastic dynamical systems. Note that despite the man-made neuroprosthesis will control a complex-living system, it shall not necessarily obey biological control principals: estimation procedures can be used to translate the brain signal into desired motor actions that can be used as a suitable command of a robot arm or other prosthetic interface. The final goal is an undeniably control, but it undoubtedly provides natural control suitable for clinically viable systems.

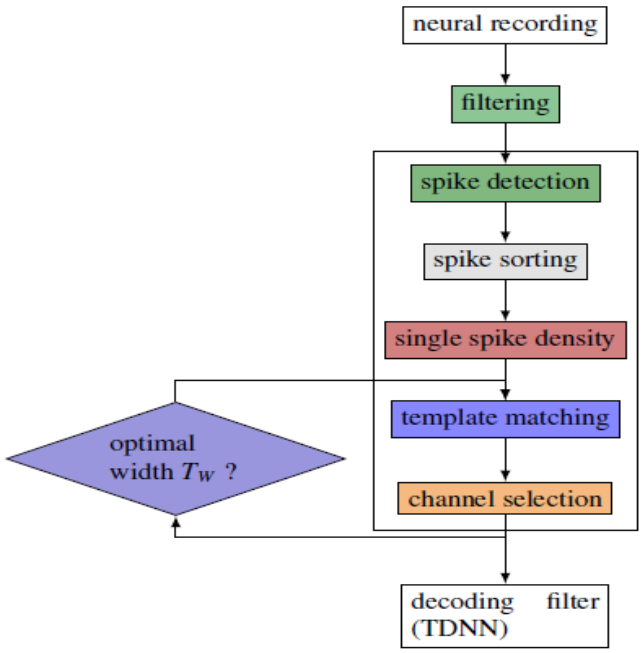


Figure 1. Plan of controlled prostheses: Intra-cortical electrode arrays are used to record neural signals

Estimation of times of action potentials choosing optimally the width of the analysis window TW and assigned to neural units. A decoder is used to translate the neural activity into control signals of the prosthetic system. In this article, we aim to design a rat's hand trajectory decoder from its brain signals by analyzing neuronal spiking activity of motor cortical recorded from two freely moving rats in the first two months after electrode implantation as exposed in section 2.1. The experiments are realized at the National TsinHua University (NTHU) in Taiwan. In section 3 is described several kinds of specific spatio-temporal models and the manner of their detection. We explore in addition the statistical approaches to evaluate the probability to optimize the pattern detection. Section 6 gives summarizing statistics on the motor cortex responses recorded simultaneously with movement to make proof of the mapping linking these models to special behavioral task using time delay neural network.

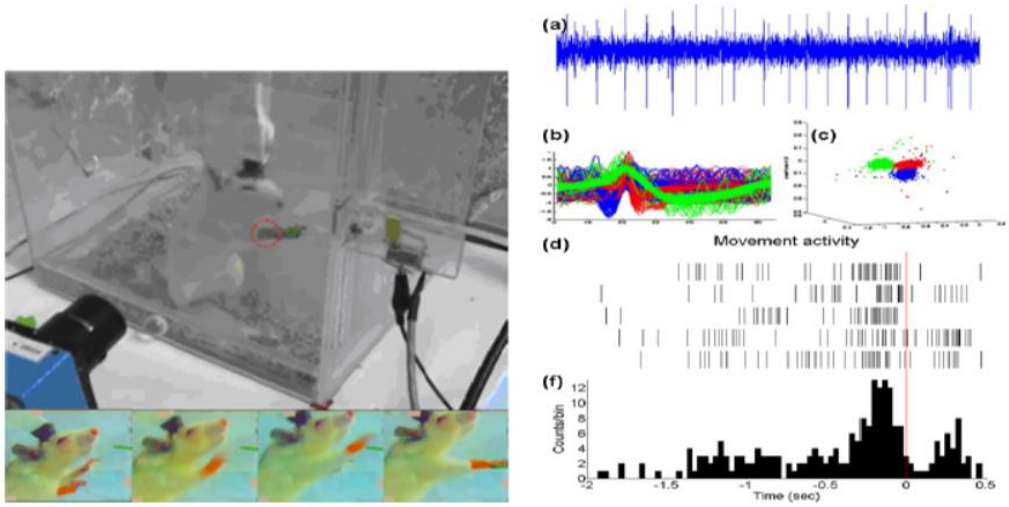
2. EXPERIMENTATION AND DATA ACQUISITION

2.1. Animal training and behavioral tasks

In The study, approved by the Use Committee at the National Chiao Tung University and Institutional Animal Care, was conducted according to the standards established in the Guide for the Care and Use of Laboratory Animals. Four male Wistar rats weighing 250-300 g (BioLASCO Taiwan Corp., Ltd.) were housed individually on a 12 h light/dark sequence, with access to food and water ad libitum. Data was recorded from the motor cortex of awake animal performing a simple reward task. In this task, male rats (BioLACO Taiwan Co., Ltd) were trained to press a lever to initiate a trial in return for a water reward as shown in Figure 2(a). The animals were water restricted 8-hours/day during training and recording session but food were always provided to the animal ad lib every day.

2.2. Chronic animal preparation and neural ensemble

Pentobarbital was used to anesthetise the animals (50 mg/kg i.p.). They were then positioned on a regular stereotaxic apparatus (Model 9000, David Kopf, USA). Electrode array was implanted after a careful dura retraction. The couples of 8 micro-wire electrode arrays (no.15140/13848, diameter of 50m; California Fine Wire Co., USA) are implanted in the primary motor cortex (M1) at the level of the layer V. The area related to forelimb movement is located anterior 2-4 mm and lateral 2-4 mm to Bregma. After implantation, the exposed brain required a week of recovery time in where it should be sealed with dental acrylic. The animal moved freely in the box to move in the box of the behavior task (30 cm_30 cm_ 60 cm) during the recording sessions, as depicted in Figure 2(a), to receive 1 ml of rewarded water, the rat should only press the lever with its right forelimb. Neural signals are logged using a Multi-Channel Acquisition Processor (MAP, Plexon Inc., USA). The recorded signals were exposed from the main stage to an amplifier, over a band-pass filter (spike pre-amp filter: 450-5 kHz; gain: 15,000- 20,000), and sampled at 40 kHz per channel as given in Figure 2(b).



(a) Protocole of ECoG collection and recording by invasives microelectrodes

(b) Spikes sorting and representation

Figure 2. (a) Experiment of neural activity recording, (b) Neural recording data and spike sorting: (a) Real brain signal recorded from M1 area, (b) Spikes detection from the brain signal, (c) Spikes clustering, (d) Results of spikes sorting, (f) Histograms of firing activity near a time of behavioral task

In the same time, the animal's activities was recorded by the video tracking system (CinePlex, Plex Inc., USA) and inspected to ensure that it was consistent for all trials included in a given analysis [24]. During a daily research session, the neural activity was frequently recorded in many sequences of short periods of two to six minutes during which the rat accomplished one behavioral task, interlaced by rest periods when no recording of activity was realized. Finally, we obtained dataset composed of 48 channels (or neurons) described by successions of '1' separated by long silences of '0'. An additional illustration was used based on a smooth by a Gaussian window applied on the rate of spikes.

3. DECODER CALIBRATION AND CLOSED LOOP CONTROLLER

A statistical model is developed to estimate the decoding process. In Figure 3 is shown a generic block plan of the neural controller which operates as a sample data feedback system whose behavior depends of an ensemble of allowed inputs and an optimization criterion. It generates the controls. The trajectories are the responses to the control signals.

The captured neural signals using micro-electrodes are exhibited to a circuit of signal processing like as amplification, band-pass filtering, and then transmitted to the level of spike localization, in which we define the times of spikes arising for each channel. The previous stage results spikes trains which is exposed to the spikes sorting stage composed of two principal parts : (i) one detects spikes (waveforms) from background noise [25] and (ii) classifies each detected spike to generate multiple single-unit spike trains [26]. The related movement information (kinematic parameters) is reconstructed using the simultaneous rates of the obtained trains of spikes. Usually, this procedure is named neuronal spike decoding [9] and it is performed by the decoding filter stage in Figure 1. The performance of this decoding filter is of great importance since it permits the quantitative information that is encoded in the spike trains to be estimated. In [27], Bialek reported the possibility of the trajectory reconstruction using a linear finite impulse response (FIR) filter applied on spikes train.

In this paper we are concerned with dynamic nonlinear discrete-time model with time-invariant parameters [28]. The parameters w are considered as random variables adjusted iteratively until the responses of the adaptive model linked better with the measured outputs in the sense specified by the minimization criterion. The signals are mostly described as parameters. The observables, or output variables, are the time-histories of the dynamic system. The statistics of the non-measured signals, e.g. the noise, are supposed to be known but one needs to estimate them based on the observation data.

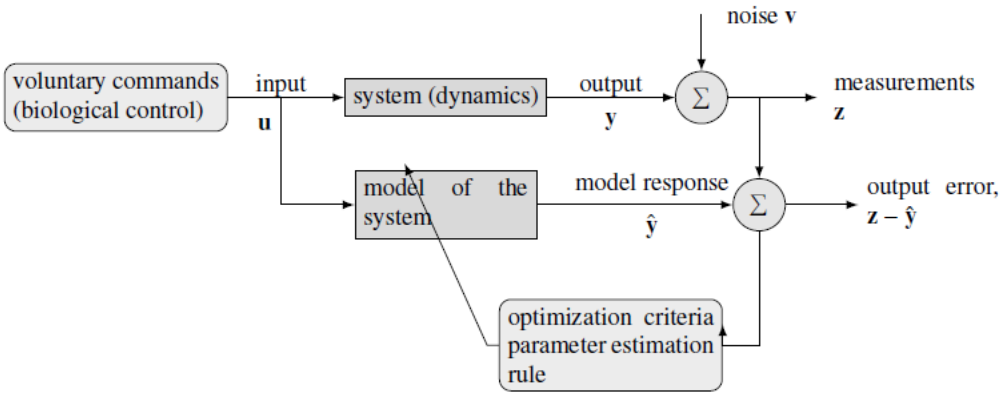


Figure 3. Block diagram representation of a nonlinear feedback control system. Decoding of position from ensemble rat neural spiking activity

4. RECURRENT NEURAL NETWORK

4.1. The NARX models

Despite that linear filters such as wiener filters [29] deliver a simple analytic solution with low complexity training and online feasibility, the reconstructed trajectories may be suboptimal since the output is limited to mappings in the input space and mapping of control features to motor behavior may include nonlinear translations. Therefore, herein, we study modeling hand movements using artificial neural networks. The NARX (Nonlinear AutoRegressive with eXogenous inputs) model is an essential type of nonlinear discrete time systems for time-series forecasting [30, 31]. The recurrent NARX can be represented:

$$y(t) = f(u(t-\Delta), \dots, u(t-n\Delta); y(t-\Delta), y(t-2\Delta), \dots) \quad (1)$$

where the regressor is represented by $u(t) = \left(u(t-\Delta), \dots u(t-n\Delta)\right)^T \in \mathbb{R}^{q \times n}$ and $y(t)$ represents the output of the system at time t; $\Delta, 2\Delta, \dots, n\Delta$ are the time lags of the system variables and $f(.)$ an unknown nonlinear function. The general prediction (1) computes the next value of time series $y(t)$ (output) from the past observation u and the past output y. Note that u is multidimensional and sized by the number q of neural channels. Regarding this way, the prediction becomes a function estimation problem, and so the aim the technique is focused to the approximation of the function f, like a multi-layer perceptron (MLP). The obtained system is a compact embedded memory model called a NARX network as sketched in Figure 4: a first tapped delay line of inputs with a delayed connections from the output to the input.

A Time Delay Neural Networks (TDNN) is a NARX with zero output-memory order [32] given by the form:

$$y(t) = f\left(u(t-\Delta), \dots, u(t-n\Delta)\right) \quad (2)$$

This formulation is simplified but with a significant restriction of the NARX from its representational abilities like a dynamic network. Simulated results given by Lee *et al*. [33] show that NARX networks are often much better than conventional recurrent neural networks at discovering long time dependences. An explanation why output delays can help long-term dependences can be found by considering how gradients are calculated using gradient-descent learning algorithms: for gradient based training algorithms, the information about the gradient contribution m steps in the past vanishes for large m, i.e. mathematically $\lim_{m \to \infty} \frac{\partial y(t)}{\partial y(t-m)} = 0 \ \forall t$ where y is the state variable (and output neuron) and t the time index.

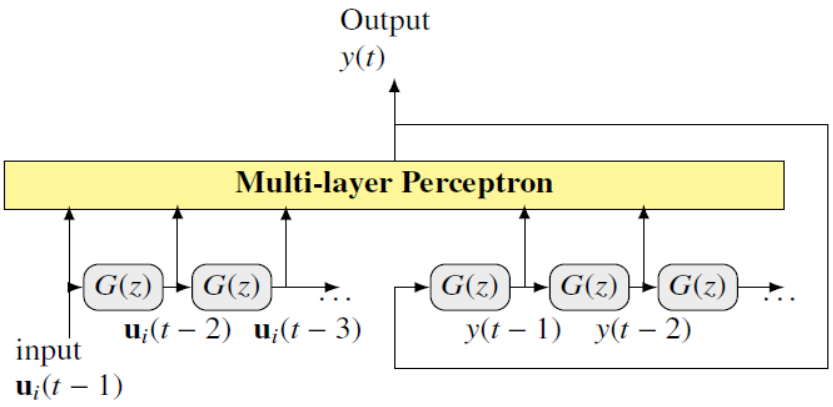


Figure 4. Architecture of the NARX network: Only the i^{th} regressor $u_i(t) = \left(u_i(t-1), \dots, u_i(t-n_i)\right)^T$ is shown (among the q possible neural channels). The inputs are fully connected to the hidden layer of the MLP. Note that the transfer functions $G(z) = z^{-1}$ reduces the structure to a tap-delay-line

4.2. Learning algorithms

The number of inputs scales the number of parameters of the model and thus creates problems with model generalization. Moreover, the size of the training data required for good approximation increases with the number of parameters. To overcome the problem of model order, a dynamic back-propagation algorithm is required to compute the gradients, which defer the memory structure to the hidden, recurrent layer instead of time-embedding but is more computationally intensive than static back-propagation and takes more time. In addition, training is more likely to be trapped in local minima [30]. Simple back-propagation algorithm is inappropriate since the parameter convergence need long iterations and a learning rate to adjust. Hence we propose to use the conjugate gradient algorithm (CGA) to accelerate the weight update. The weight vector of a feed-forward neural network is a point in the real Euclidean space $\mathbb{R}^N$ defined by

$$w = \left(\dots, w_{i,j}^{(l)}, w_{i+1,j}^{(l)}, \dots, w_{n_l,j}^{(l)}, \dots\right)^T \quad (3)$$

where $w_{i,j}^{(l)}$ is the weight from unit i in the layer l to unit j is layer $l+1$, $n_l$is the number of units in layer l. We assume a global error function $E(w)$ depending of the weight and biases attached to the network and chosen here as the standard least square function.

$$E(w) = \sum_{t=1}^{T} \left(y(t) - f(u(t-1), \dots, u(t-n); y(t-1), y(t-2), \dots; w)\right)^2 \quad (4)$$

$E(w)$ is evaluated with one pass forward and the gradient $E'(w)$ with one pass backward:

$$E' = \left(\dots, \sum_{t=1}^{T} \frac{\partial E(t)}{\partial w_{i,j}^{(l)}}, \sum_{t=1}^{T} \frac{\partial E(t)}{\partial w_{i+1,j}^{(l)}}, \dots, \sum_{t=1}^{T} \frac{\partial E(t)}{\partial w_{n_l,j}^{(l)}}, \dots \right)^T \quad (5)$$

Back propagation update rule is given by

$$w^{(k+1)} = w^{(k)} - \alpha_k E'\left(w^{(k)}\right) \quad (6)$$

At a time instance k, where $-E'\left(w^{(k)}\right)$ is the gradient vector and $\alpha_k$is the learning rate that should be chosen carefully to make the algorithm is based on the linear approximation $E(w+h) \approx E(h) + E'(w)^T h$, the algorithm show poor convergence. Hence, a Conjugate Gradient Algorithm (CGA) is generally a better choice and results in a faster convergence to reach the minimum of a $E(w)$. The optimization strategy consists to choose a search direction p_k and then to decide how far to go in the specified direction, i.e. to determine the step size α_k.CGA chooses the search direction and the step size by using second-order approximation.

$$E(w+h) \approx E(w) + E'(w)^T h + \frac{1}{2} h^T E''(w) h \quad (7)$$

where h is the line search direction [34]. From the current search direction, the next search direction is determined so that it is conjugate to previous search directions. The critical points of $E(w)$ are the solutions of the linear system (8).

$$E'_q(h) = E''(w)^T h + E'(w)^T = 0 \quad (8)$$

If $E'_q(h)$ denote the quadratic approximation to E in the neighborhood of a point w so that $E(w+h) = E_q(h)$. Let the vector basis $p_1, \dots, p_N$ be a conjugate system of $\mathbb{R}^N$, the step from a starting point h_1 to a critical point h^*can be expressed as a linear combination of $p_1, \dots, p_N$

$$h^* - h_1 = \sum_{i=1}^{N} \alpha_i p_i \; \alpha_i \in \mathbb{R} \quad (9)$$

Multiplying (9) with $p_j^T E''(w)$ and substituting$E'(w)$ for $-E''(w)^T h^*$ from (8) gives:

$$p_j^T(-E'(w) - E''(w)h_1) = \alpha_j p_j^T E''(w) p_j \quad (10)$$

$$\Rightarrow \alpha_j = \frac{p_j^T(-E'(w) - E''(w)h_1)}{p_j^T E''(w) p_j} = \frac{-p_j^T E'(h_1)}{p_j^T E''(w) p_j} \quad (11)$$

The intermediate points $h_{k+1} = h_k + \alpha_k p_k$ given the iterative determination of $h^*$are in fact minima for $E(h)$ restricted to the plane P_k described by $h = h_1 + \alpha_1 p_1 + \cdots + \alpha_k p_k$. The conjugate weight vector $p_1, \dots, p_N$ are determined recursively:

$$p_{k+1} = r_{k+1} + \beta_k p_k \quad (12)$$

where

$$r_{k+1} = -E'(h_{k+1}),\ \ \beta_k = \frac{|r_{k+1}|^2 - r_{k+1}r_k}{p_k^T r_k} \tag{13}$$

and $h_{k+1} = k_k + \alpha_k p_k$, $\alpha_k = \mu_k/\delta_k$, $\mu_k = p_k^T E_q'(h_k)$ and $\delta_k = p_k^T E''(w)p_k$. For each iteration the above described algorithm is applied to the quadratic approximation E_q of the global error function. In most cases, preconditioning is necessary to ensure fast convergence of the conjugate gradient method that takes the following form in algorithm 1.

Algorithm1 Scaled CGA
Require: choose initial weight vector w^1 (can be 0);
Ensure: $r^{(1)} = p_1 - E'(w^{(1)})$;
1: $k \leftarrow 1$
2: repeat
3: Calculate second-order information:
$s_k = E''(w_k)p_k$
$\delta_k = p_k^T s_k$
4: Calculate step size:
$\mu_k = p_k^T r_k$
$\alpha_k = \frac{\mu_k}{\delta_k}$
5: Update weight vector:
$w_{k+1} = w_k + \alpha_k p_k$
$r_{k+1} = -E'(w_{k+1})$
6: If k mod $N = 0$ then restart algorithm: $p_{k+1} = r_{k+1}$
Else create new conjugate direction:
$\beta_k = \frac{|r_{k+1}|^2 - r_{k+1}r_k}{\mu_k}$
$p_{k+1} = r_{k+1} + \beta_k p_k$
7: until the steepest descent direction $r_k \approx 0$; then set $k = k+1$ and go to 2
Else terminate.
8: return w_{k+1}

The conjugate gradient algorithm works very well on well-conditioned matrices; however, in reality, most matrices are ill-conditioned, reducing the efficiency of the algorithm. Môller showed that algorithm 1 might restart when $r_{k+1} \approx r_k$ or $p_k \approx r_{k+1}$. Hence he proposed in [35] different versions of the conjugate gradient methods in regulating indefiniteness of $E''(w_k)$ with Lagrange multiplier λ_k. This is done by setting

$$s_k = \frac{E'(w_k + \sigma_k p_k) - E'(w_k)}{\sigma_k} \tag{14}$$

and for each iteration adjusting λ_k looking at the sign of δ_k. This results in the following algorithm 2

Algorithm1 Preconditioned CGA
Require: choose initial weight vector w^1 (can be 0) and scalars
$\sigma = 0,\ \lambda > 0$ and $\bar{\lambda} = 0$;
Ensure: $\boldsymbol{r^{(1)} = p_1 - E'(w^{(1)})}$;
1: $k \leftarrow 1$ and $success = true$
2: **if** $success = true$ **then**
3: **repeat**
4: Calculate second-order information:
$\sigma_k = \frac{\sigma}{|p_k|} s_k = \frac{E'(w_k+\sigma_k p_k) - E'(w_k)}{\sigma_k} \delta_k = p_k^T s_k$
5: Scale s_k :
$s_k = s_k + (\lambda - \bar{\lambda}_k)p_k \delta_k = \delta_k + (\lambda - \bar{\lambda}_k)|p_k|^2$
6: **if** $\delta_k \leq 0$**then**
make the Hessian matrix positive definite:
$s_k = s_k + \left(\lambda_k - 2\frac{\delta_k}{|p|^2}\right) p_k \bar{\lambda}_k = 2\left(\lambda_k - 2\frac{\delta_k}{|p|^2}\right)\delta_k = -\delta_k + \lambda_k |p|^2 \lambda_k = \bar{\lambda}_k$
end if
7: Calculate step size:
$\mu_k = p_k^T r_k \alpha_k = \frac{\mu_k}{\delta_k}$
8: Calculate the comparison parameter:
$\Delta_k = \frac{2\delta_k\left(E(w_k) - E'(w_k + \alpha_k p_k)\right)}{\mu_k^2}$

```
9:   if Δ_k ≥ 0 then
     a successful reduction in error can be made:
        w_{k+1} = w_k + α_k p_k r_{k+1} = −E′(w_{k+1})
10:  if k mod N = 0 then restart algorithm: p_{k+1} = r_{k+1}
11:  else create new conjugate direction:
        β_k = (|r_{k+1}|² − r_{k+1} r_k) / μ_k   p_{k+1} = r_{k+1} + β_k p_k
     end if
12:  if Δ_k ≥ 0,75 then
     reduce the scale parameter: λ_k = λ_k/2
13:  else
     a reduction in error is not possible : λ̄_k = λ_k
     success = false
     end if
14:  if Δ_k ≤ 0,25 then increase the scale parameter: λ_k = 4λ_k
     end if
     end if
15:  until the steepest descent direction r_k ≈ 0; then set k = k + 1 and go to 2 else terminate.
8:   return w_{k+1}
end if
```

MLPs trained by a back-propagation algorithm were sometimes employed for the BMI (see e.g. Wessberg et al. [36] or Wang [37]), however, some difficulties were appeared in the training process. Like the number of parameters, which refers to the question of the amount of connections or weights contained the network furthermore the optimally selected the inputs. In the linear case, several approaches have been proposed in the orders determination (see [38] for review): penalizing the parameter increase (see [39] and [40] for review), delay-damage algorithm [33], (subset) feature selection [38] or extraction [41], etc. These answers are part of a more general question: How time is embodied in temporal patterns?

5. HOW TIME EMBODIED IN TEMPORAL PATTERNS?

5.1. Time feature selection in neural channels

Model dimension reduction can improve the precision of its predictive inference because, in high dimensions, prediction or inference may be computationally costly. Moreover, the reduction of features can offer simple visions onto the function of the system. Temporal processing of neural data is a challenge because the information is embedded in (i) the temporal order which refer to the ordering among the components of a sequence and by (ii) the time duration, i.e. the time that separates two time stamps [29].

Generally, selections of time lag for the prediction setting and of the best variables selection are done manually by experts. Selection methods of input variable can be classified into two classes: methods of filtering and wrapper methods (see Guyon and Elisseeff [42] for a review). Filter methods use statistical measures to classify the variables, according to their influence and relevance on the target variable. On the other hand, wrapping methods use the learning model as the basis for selection. Notice that the latter may often achieve more accurate prediction results because variables selection will take into account the approximation model. Correlation method, partial correlation method, Mallows coefficient and PLS based method are often used tor time-lag selection. Shakil et al. propose a linear time-lag model optimized by a genetic algorithm for delay selection [43]. Performing the selection of the time-lags for each variable before variable selection improves the final prediction performance.

For a better knowledge about the variables that affect the target variable y, the best choice is by means of filter methods because they use only the information content of data rather than methods based on correlation coefficient. But for systems involving nonlinear interactions among variables, linear filter methods fail. The linear correlation value which can measure nonlinear interaction among candidates of input variables and the target output is a better choice [41]. But the main difficulty lies in the estimation of the mutual information in multidimensional space which is computationally expensive in addition [44]. Building a predictor from a selection of the most relevant variables is usually suboptimal; conversely a subset of useful variables may exclude many redundant, but relevant, variables. This contrasts with the problem of ranking all potentially relevant variables.

5.2. Basic elements in the decoding algorithm

Neuronal assemblies contribute to neural decoding. Accordingly, synchronized firing spikes are considered a property of neuronal signals that can be detected and depends on stimulus and behavioral context. There is a need for analysis tools that allow us to reliably detect correlated spiking activity between multiple neurons that is not explained by the firing rates of the neurons alone.

First, the algorithm performs channel per channel coarse preliminary spike detection to estimate the spike morphology. The raw signal $s(t)$ is raised to the power: $s_r(t) = |s(t)|^r$, with $r = 2$ but higher values may improve the performance in increasing the influence of large-amplitude events. Then we apply a Blackman low-pass filter to $s_r(t)$ with a cut-off frequency in the range of 100-250Hz to decrease the effect of noise at high frequencies and to reduce the number of signal maxima that will be considered as candidate spikes. We then take the r^{th} root of the filtered signal to get the final output signal for this stage $x(t) = [h(t) * s_r(t)]^{\frac{1}{r}}$, where $h(t)$is the FIR impulse response of the low-pass filter and (*) the convolution operator. After the transformation, $x(t)$ presents a sharp increase for an occurrence of spike and a soft maxima increase when no spike appears (e.g. noise). We denote the sequence of spike arrival times i.e. $x(t)$ maxima as $t_0 \leq t_1 \leq t_2, \dots, t_k \leq t_{k+1}, \dots, \leq t_k \leq T$ and the amplitude of the maxima as $\{x_1, \dots, x_k\}$ where $x_i = x(t_i)$ and K is the number of maxima in $x(t)$.

After time discretization, simultaneously recorded spiking activities are represented as simultaneous sequences of ones and zeros; "1" is used to indicate the presence of a spike and "0" for the nonappearance of spike as shown in Figure 5. There are at most 2^q different coincidence patterns with q simultaneously observed neurons (in fact much lower since zeros dominate). The function of the probability density of these maxima has two different prominent styles: a mode for ECoG background noise close to zero and a mode distant from zero due to the spike activity, which can be easily separated with a threshold calculated by a simple neyman-pearson test, see [45] for a reminder on signal detection. After selecting the threshold, the set $T = \{\tau_1, \dots, \tau_{k'}\}$ represents the times of detected spikes $(K' < K)$. The latter are used to estimate the q templates $\Gamma_i, i = 1, \dots, q$ by a simple mean; we estimate these time range span templatesΓ_i between 0,45ms and 0,55ms before and after the peak of the template.

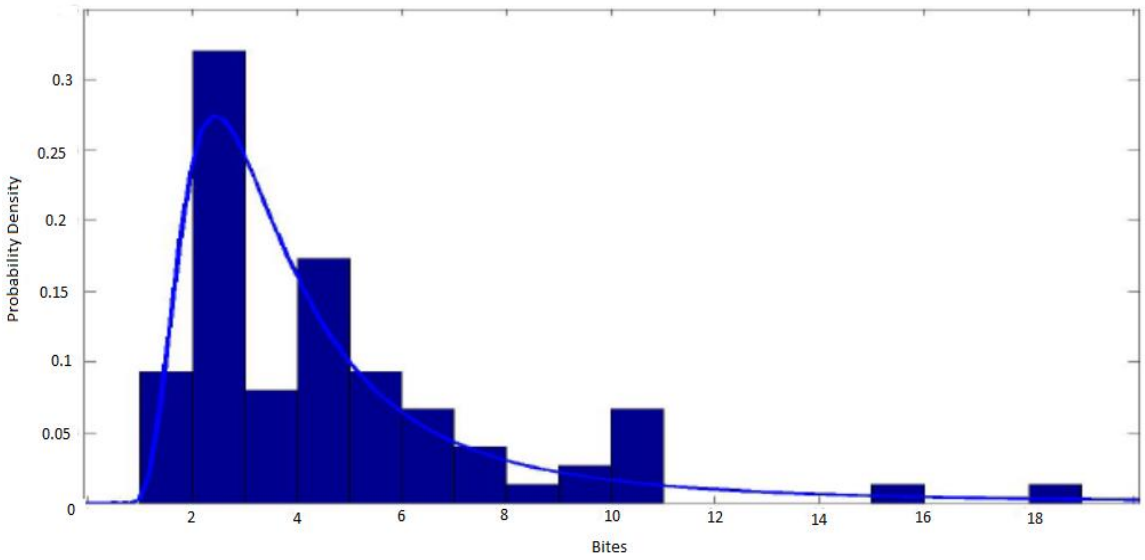
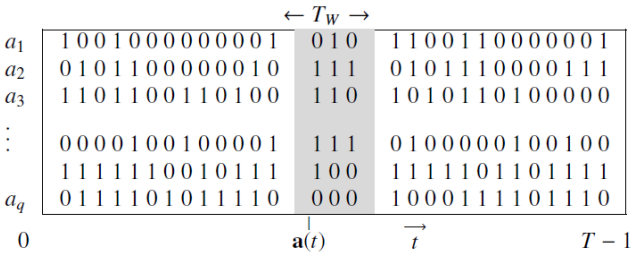


Figure 5. The spiking activity of q instantaneously recorded neuron is encoded as a binary representation of q parallel spike trains within a data stretch of T bins

The normalized cross-correlation, used for finding matches of the reference template Γ is obtained by sliding the template window of size J:

$$m(t) = \frac{\Gamma^T x}{\sigma(\Gamma^T \Gamma)^{1/2}} \tag{15}$$

$m(t)$ is scaled by the variances of the template and the observed signal, wit $x = \left(x\left(t - \frac{T_W}{2}\right), \dots, x\left(t + \frac{T_W}{2} - 1\right)\right)^T, \Gamma = \left(\Gamma(1), \dots, \Gamma(J)\right)^T$. The width of the analyses window T_W (expressed in number of bins) is chosen to optimally identify excess synchrony as significant. Typically, the sample size is $T^W \leq J$.



– q Parallel binary processes, lasting for T time steps. Each row consists of a sequence of 1s and 0s representing a realization of a single process a_i. The 1s mark the occurrences of spike events and the vector $a(t)$expresses the joint activity across the process. Correlation is captured sliding a window analysis through a binary data representation of the events.
– The observed number of patterns is governed by a binomial distribution of k neurons firing jointly.

It is useful to think of Γ as a (known) signal originating from a source: the measurement vector is $x = \mu\Gamma + \sigma n$ and the question is whether $\mu = 0$ or $\mu > 0$. n is seen as a noise vector scaled by a constant σ and added to the signal. The detection problem is phrased as $H_0{:}\mu = 0$ versus $H_1{:}\mu > 0$ in the model $x \sim \mathcal{N}(\mu\Gamma, \sigma^2 I)$. The distribution of the statistic $m(t)$ is normal $x \sim \mathcal{N}\left(\frac{\mu}{\sigma}(\Gamma^T\Gamma)^{1/2}, I\right)$. This means that we can set a threshold m_0 for testing H_0 vs H_1:

$$\phi(x) = \begin{cases} 1, m > m_0 \ \ under\ H_1 \\ 0, m \le m_0 \ \ under\ H_0 \end{cases} \tag{16}$$

The threshold $m_0$to be determined is computed so that the false alarm probability PFA is minimized and the detection probability PD is maximized. The detection probability is determined by the formula

$$PD = \int_{m_0}^{\infty} (2\pi)^{-1/2} \exp\left(-\left[x - \frac{\mu}{\sigma}(\Gamma^T\Gamma)^{\frac{1}{2}}\right]^2 /2\right) dx \tag{17}$$

$$= 1 - \phi\left(m_0 - \frac{\mu}{\sigma}(\Gamma^T\Gamma)^{1/2}\right) \tag{18}$$

and the false alarm probability by $PFA = 1 - \phi(m0)$. Fixing in our experiment $PFA = 10\%$, we obtain $m0 = 1.28$.

Data are organized and aligned with the corresponding behavioral event or stimulus into “trials”. Figure 6 gives the detection probability PD with times for a set of selected channels: the plots suggest that the patterns of variation are very similar and the series are positively correlated at lag 0, suggesting synchrony.

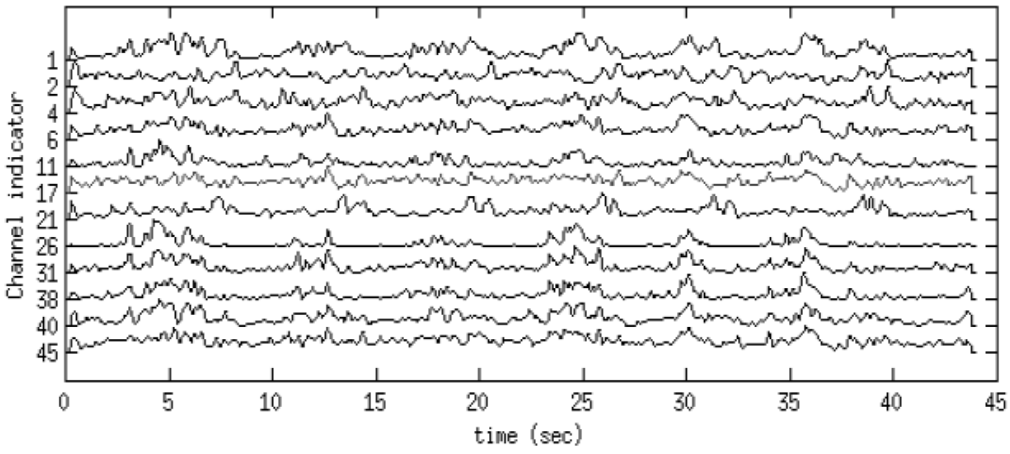


Figure 6. Detection probability PD through times when fixing the false alarm probability to 10%

Finally, we calculate the performance of these signals detection technique by computing sensitivity and specificity of the channels for a range of descriptors. Table 1 presents the ten "best" neural channels for which six descriptors have been computed: the template variance and mean resp. μ_i and σ_i, the variation coefficient $cv_i = \frac{\sigma_i}{\mu_i}$, the early firing time before the behavior event is detected Δt_i, the maximum rate of missing events $ME_i$and the maximal amplitude of the template MA_i. The ideal channel is a channel with a detection time Δt_ias large as possible, a small variability $cv_i$and a low missing detection rate ME_i.

This electrode selection strategy, originally suggested in [41], was based on our motivating rationale that channels eliminated before the main processing would further simplify the decoding of motor cortical recordings. Our experiment retain, in the following, 4 channels at maximum.

Table 1. Channel ranking obtained from a range of descriptors computed on individual channels. The two last channels are randomly chosen from the non-selected group of channels: there are no time detection because of the non-regularity of templates

ith ch	σ_i	μ_i	cv_i	Δt_i	ME_i	MA_i
21	**0.2090**	0.8530	**0.2450**	**0.500**	**0.1930**	6.6144
25	0.2575	0.8792	0.2928	0.360	0.4044	7.8553
43	0.2395	0.8108	0.2954	0.280	0.3116	4.1574
2	0.2753	0.9088	0.3029	0.468	0.3911	5.8860
42	0.2849	0.8688	0.3279	0.250	0.4095	**7.9383**
29	0.2800	**0.7923**	0.3534	0.365	0.4985	4.1551
1	0.3524	0.9324	0.3779	0.299	0.5307	5.8253
22	0.3202	0.8095	0.3955	0.430	0.4298	6.4084
14	0.3391	0.7997	0.4241	0.350	0.4638	2.6330
5	2.0931	0.4095	3.3851	--	5.6899	0.2919
13	2.3192	0.4407	1.6367	--	5.5854	0.2598

5.3. Metaneuron and nonparametric optimal detector of joint-spike event

The instantaneous observations of spike actions from q channels can be described by parallel binary sequences $a_k(t) \in \{0,1\}$ and the q sequences as a vector-valued function $a(t)$ formed by the $a_k(t)$: $a(t) = \left(a_1(t), \dots, a_q(t)\right)^T$. Algorithm 3 summarizes the main features of the previous processing.

```
Algorithm3 Joint-Spike Detection
1: Align trials, decide on width of analysis window.
2: Decide on allowed coincidence width
3: Perform a sliding window analysis
4: for each window do
5: Detect and count coincidences
6: Calculate expected number of coincidences
7: Test significance of detected coincidences
8: end for
9: Explore behavioral relevance of coincidence epochs.
```

The input to the detector is formed by the vector $a(t)$. The simplest decision rule takes the following form:

$$\sum_{i=1}^{q} a_i \lessgtr \theta \tag{19}$$

The decision (19) counts the number of $1s$ in a sliding window. Suppose that the a_i's are independent but not necessarily iid [46]. The detection problem may be written as testing H_0versus H_1. When H_0 is true, $P(a|H_1)$ has a zero median whilst if H_1 is true, $P(z|H_1)$ has a positive median. So the test can be stated in terms of a_i as

$$P(a_i|H_0) = \begin{cases} \frac{1}{2} a_i = 0 \\ \frac{1}{2} a_i = 1 \end{cases} \quad P(a_i|H_1) = \begin{cases} 1 - fa_i = 0 \\ fa_i = 1 \end{cases} \tag{20}$$

The likelihood ratio can now be formed and compared to a threshold λ:

$$\Lambda(a) = \frac{P(a|H_1)}{P(a|H_0)} = \frac{f^{\sum_{i=1}^{q} a_i}(1-f)^{q-\sum_{i=1}^{q} a_i}}{\left(\frac{1}{2}\right)^q} \underset{d_2}{\overset{d_1}{\lessgtr}} \theta \tag{21}$$

Let $q^+ = \sum_{i=1}^{q} a_i$, the number of $1s$, then $\Lambda(a)$ becomes

$$\Lambda(a) = \frac{f^{q^+}(1-f)^{q-q^+}}{\left(\frac{1}{2}\right)^q} = [2(1-f)]^q \left(\frac{f}{1-f}\right)^{q^+} \tag{22}$$

Taking the natural logarithms of both sides and after reduction we obtain

$$q^+ \underset{d_2}{\overset{d_1}{\lessgtr}} \theta - f \ln_{\left(\frac{f}{1-f}\right)}[2(1-f)] = \theta' \tag{23}$$

The test in (23) compares the number of $1s$ with the threshold θ'. The threshold is fixed by setting the false alarm probability $P(d_2|H_0)$ (as in section 5.2) to be less than or equal to $\alpha = 10\%$. Since $q^+$is the sum of q Bernoulli random variables, q^+ obeys binomial distribution $q^+ \sim \mathcal{B}\left(q, \frac{1}{2}\right)$. [47]. If H_0 is true

$$P(q^+ = k|H_0) = C_q^k \left(\frac{1}{2}\right)^k \left(1 - \frac{1}{2}\right)^{q-k} = C_q^k \left(\frac{1}{2}\right)^q \tag{24}$$

where the binomial coefficient is $C_q^k = \frac{q!}{k!(q-k)!}$. The false alarm probability is therefore

$$P(q_2|H_0) = P(q^+ > \theta'|H_0) = \sum_{k=\theta'+1}^{q} C_q^k \left(\frac{1}{2}\right)^q \tag{25}$$

We can find the smallest value of θ' such that $P(d_2|H_0) \leq \alpha$ or

$$\sum_{k=\theta'+1}^{q} C_q^k \left(\frac{1}{2}\right)^q \leq \alpha \tag{26}$$

Once θ' is fixed, one can easily derived the probability of detection (POD) from the binomial distribution $\mathcal{B}(q, f)$, and this yields

$$P(q^+ = k|H_1) = C_q^k f^k (1-f)^{q-k} \tag{27}$$

Therefore the POD is

$$P(d_2|H_1) = P(q^+ \geq \theta'|H_1) = \sum_{k=\theta'+1}^{q} C_q^k f^k (1-f)^{q-k} \tag{18}$$

Finally the non-parametric decision rule is easy to implement and requires that we sum the components of the vector $\mathrm{a}_k(t)$. As an illustration, let us consider a case with $q = 10$ channels and $\alpha = 25\%$. The threshold θ' for this test is obtained from (26) by finding the value of θ' for which

$$\sum_{k=\theta'+1}^{10} C_{10}^k \left(\frac{1}{2}\right)^{10} \leq 0.25 \tag{29}$$

The desired binomial coefficients are:

$$C_{10}^{10} = 1 \quad C_{10}^{9} = 10 \quad C_{10}^{8} = 45 \quad C_{10}^{7} = 120 \quad C_{10}^{6} = 210$$

Hence, if we let $\theta' = 6$, we have $\sum_{k=7}^{10} 1 + 10 + 45 + 120 = 176 \leq 256$. Therefore, the test for this problem is given by $q^{+} = \sum_{k=1}^{10} a_i \lessgtr 6$. For $\theta' = 6$, the actual value of the false alarm probability is $PFA = \sum_{k=7}^{10} C_{10}^{k} 2^{-10} = 176/1024 = 0.17$. Suppose that we obtain the following set of observations $\mathrm{a} = (0111110010)^{T}$. For this set of observations, $q^{+} = 6$ and we decide d_1.

6. PERFORMANCE ASSESSMENT

6.1. Experiment assessment with one neuron

First, we start our simulation with a single input as in a similar work presented in [48], where the best channel is selected with the correlation method (channel 21) which was the unique input to the TDNN but with a try error time regression. In this time, the best selected channel is given using the variation coefficient cv and the time shifting is defined by the template matching. Table 2 gives the time training and its log, and the estimation error with the corresponding neurons in the hidden layer. A single input channel with different neurons of the hidden layer, respectively (2; 4; 7; 9; 12; 15; 20; 25), is used the training time and the estimation error are given in Table 2 below with respect to the number of neurons.

Table 2. Results simulation with one input

Neurons	log(time)	time(sec)	error(mm)
2	4.8114	122.8988	8793
4	5.0279	152.6142	2779.2
7	5.4143	224.6053	28.4411
9	5.6386	281.0812	0.7123
12	5.9613	388.1077	0.7656
15	6.2793	533.4191	0.2185
20	6.7274	834.9931	0.4679
25	7.0821	1190.5	0.6063

6.2. Experiment with multiple neurons

For multiple neurons (channels), we use the first two and three selected neurons with respect to their order.

6.2.1. Two inputs

For the double neurons, we started our simulation using the time delay neural network with (2; 4; 7; 9; 12; 15; 20; 25; 30) neurons. The best selected neurons are the two channels 21 and 25 keeping their order. In Table 3 is given the training time of the TDNN.

Table 3. Results simulation with two inputs

Neurons	log(time)	time($10^3 s$)	error(mm)
2	4.8519	0.1280	0.0697
4	4.9234	0.1375	0.3333
7	6.3729	0.5858	0.0979
9	6.6495	0.7724	0.2668
12	7.0341	1.1347	0.2020
15	7.6010	2.0002	0.2133
20	8.1951	3.6232	0.2535
25	8.6910	5.9490	0.2927
30	9.0956	8.9161	0.0989

6.2.2. Three inputs

For this case, we have three selected inputs which are the past two selected for the case of double input with the channel 43 in the third rank. The TDNN was trained on half data and tested on the rest; in Table 4 are given results. We obtained 48 channels ECoG after spikes detection and sorting, the obtained signal is represented by suits of '1' pulses disjointed by long silences of '0'. Various methods of data illustration were suggested to perform the information hidden in distances between spikes. A brief description of these methods is summarized in [49].

Table 4. Results simulation using three inputs

Neurons	log(time)	time($10^3 s$)	error(mm)
2	4.8519	0.0129	0.2713
4	5.4893	0.0242	0.0847
7	6.8829	0.0975	0.4108
9	7.4274	0.1681	0.0814
12	7.9518	0.2841	0.4487
15	8.4373	0.4616	0.1535
20	9.1152	0.9092	0.3883
25	9.6679	1.5803	0.1362
30	10.1438	2.5432	0.1375

In this work, we compute the spike rate using a sliding window of the same frequency of the used camera get out, simultaneously, the hand coordinates with the brain signal. We calculated and used correlation coefficients to choose channels which will be used as features of the brain machine interface decoder and also to remove the redundant and non-correlated input variables with the output. Best cross-correlation values give the delay of the explanatory variables that will be used as network inputs.

In Figure 4 is shown the TDNN model. The input layer comprises, in this experiment, 10 delayed cortical signals from the 3 most substantial neurons (among 46 neurons having been recorded), between 2-7 hidden neurons with hyperbolic tangent cell function. In section 4.2, is described how we trained the TDNN using the 2 algorithms. The output variables (hand position) are centered and reduced to follow a Gaussian law $\mathcal{N}(0;\ 1)$. The layer weights are initialized randomly in $[-1;\ +1]$. We set the learning rate $\eta = 0.05$ in both algorithms.

The training was terminated when the cross validation error continuously increased for more than 10 steps. The network specifications are listed in Table 5. While the choice for the right architecture is mainly intuitive and implies arbitrary decisions, an attempt to apply ANN directly fails due to the dimensionality of the inputs. Therefore the dimension of the inputs has been reduced drastically by feature selection.

Table 5. Average AIC

Architectures	#parameters	CGA		Pre-conditioned CGA	
		x	y	x	Y
27-2-2	62	3.691	4.723	2.700	3.720
27-3-2	92	3.394	**0.426**	**1.406**	3.435
27-4-2	122	**2.732**	2.822	2.762	**2.799**
27-5-2	152	5.066	5.148	3.089	5.132
27-6-2	182	5.105	6.162	8.117	6.133
27-7-2	212	6.531	9.570	11.541	9.550

The change in the number of units in the TDNN affects the decoding performance as shown in Table 5, the mean AIC across the 50 data sets is used to quantify the performance of each model for the two decoding algorithms. For clarity here, we have 7 peaks which represent 10% of data. In Figure 7(a), by the red line is described the model output. It can be seen that the hand coordinates on a plan is tracked and the trajectory is well reconstructed along the experiences. When movements is sufficiently complex (presence of peaks), the algorithm show the ability to catch well these peaks as shown a zoom onto the fit in Figure 7(b-g).

The Akaike information criterion (AIC) is used for the measurement of performances. As mean for model selection the AIC is used. The AIC is calculated $AIC = 2p - 2\ln(L)$, where p represents the number of model's parameters, and L denotes the maximized value of the likelihood model's function. AIC deals with the compromise between the quality of the model's fit and its complexity. The best model is characterized with a lowest AIC value.

When using in decoding a reasonable number of units (more than three), the preconditioned CGA shows its superiority to the classic CGA. Figure 7: illustrates the result obtained from the MLP 27-5-2, where the trial-error technique is used for the MLP training. The performance of the MLP 27-5-2 was much worse than that of the CGA. The performance differences between the CGA and preconditioned CGA were larger above 5 units.

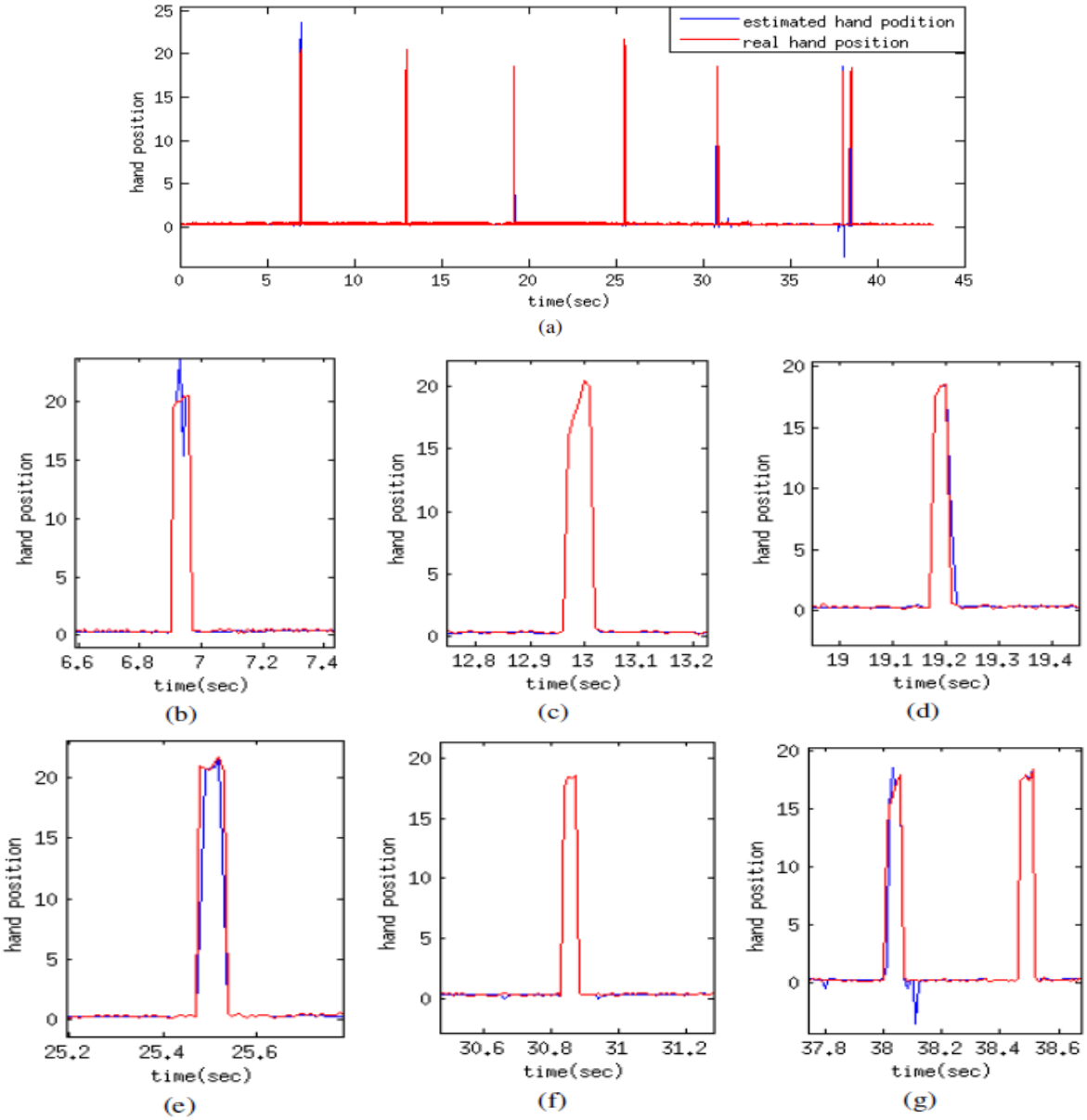


Figure 7. TDNN for hand position tracking: (a) Desired output (in blue) and model output (in red), (b-g) all peaks detection and formulation and details

7. CONCLUSION

Determining a dynamical neural network with the appropriate architecture is not easy yet critical task. Nonlinear architectures are not the only ones that can affect the performance, but also the memory architecture of dynamic models can also have an important impact on its dynamical behavior. Using the NARX network in Brain Computer Interfaces produces too many free parameters: even if the method realizes faster with the conjugate gradient algorithm. A NARX network with 20 neurons in the hidden layer and 10 times delayed of 46 inputs of ECoG brain signal yields more than 1,160 free parameters with only 10,000 samples for training and cross-validation. We proposed in this paper an algorithm to define the embedded memory order of NARX recurrent neural networks. We also show that this algorithm can demonstrate improved performance on inference tasks. Furthermore, optimizing the memory architecture and the nonlinear function through input selection often results in sparsely connected architectures but with long time windows which are able to model the global features of the underlying system quite efficiently. Simulation results from experimental brain signals showed improved performance in the hand trajectory decoding process. A perspective issue is the possibility of using in addition a Kalman pre-filter to detect pulses and group them, therefore simplify the activation measurement.

REFERENCES

[1] Kim, K., Kim, S., and Kim, S., "Superiority of nonlinear mapping in decoding multiple single-unit neuronal spike trains: A simulation study," *Journal of Neuroscience Methods,* vol. 150, no. 2, pp. 202-211, 2006.

[2] Donoghue, J., Nurmikko, A., Black, M., and Hochberg, L., "Assistive technology and robotic control using motor cortex ensemble based neural interface systems in humans with tetraplegia," *Journal of Physiology,* vol. 579, no. 3, pp. 603-611, 2007.

[3] Limousin, P., *et al*., "Electrical stimulation of the subthalamic nucleus in advanced Parkinson disease," *New England Journal of Medicine,* vol. 339, no. 16, pp. 1105-1111, 1998.

[4] Schwartz, A., "Cortical neural prosthetics," *Annu Rev Neurosci,* vol. 27, pp. 487-501, 2004.

[5] Salisbury, D. B., Parsons, T. D., Monden, K. R., Trost, Z., and Driver, S. J., "Brain-computer interface for individuals after spinal cord injury," *Rehabilitation Psychology,* vol. 61, no. 4, 2016.

[6] Bhatti, P., and Wise, K., "A 32-site 4-channel high-density electrode array for a cochlear prosthesis," *IEEE Journal of Solid-State Circuits,* vol. 41, no. 12, pp. 2965-2973, 2006.

[7] Rajangam, S, *et al*., "Wireless cortical brain-machine interface for whole-body navigation in Primates," *Scientific reports,* vol. 6, no. 1, pp. 1-3, 2016.

[8] Arzak M, Sunarya U, and Hadiyoso S., "Design and implementation of wheelchair controller based electroencephalogram signal using microcontroller," *International Journal of Electrical and Computer Engineering (IJECE),* vol. 6, no. 6, pp. 2878-2886, 2016.

[9] Brown, E., Frank, L., Tang, D., Quirk, M., and Wilson, M., "A statistical paradigm for neural spike train decoding applied to position prediction from ensemble firing patterns of rat hippocampal place cells," *The Journal of Neuroscience,* vol. 18, no. 18, pp. 7411-7425, 1998.

[10] Green, S., and Rotter, S., "Analysis of parallel spike trains," *Springer Science & Business Media,* vol. 7, 2010.

[11] Takahashi, S., Anzai, Y., and Sakurai, Y., "Automatic sorting for multi-neuronal activity recorded with tetrodes in the presence of overlapping spikes," *Journal of Neurophysiology,* vol. 89, no. 4, pp. 2245-2258, 2002.

[12] Wessberg, J., *et al*., "Realtime prediction of hand trajectory by ensembles of cortical neurons in primates," *Nature,* vol. 408, no. 6810, pp. 361-365, 2000.

[13] Dickey, A., Suminski, A., Amit, Y., and Hatsopoulos, N., "Single-unit stability using chronically implanted multielectrode arrays," *Journal of neurophysiology,* vol. 102, no. 2, pp. 1331-1339, 2009.

[14] Vigneron, V., and Chen, H., "Sparse data analysis strategy for neural spike classification," *Computational Intelligence and Neuroscience,* 2014.

[15] A. B. M. Aowlad Hossain, Md. Wasiur Rahman, and Manjurul Ahsan Riheen, "Left and right hand movements EEG signals classification using wavelet transform and probabilistic neural network," *International Journal of Electrical and Computer Engineering (IJECE),* vol. 5, no. 1, pp. 92-101, 2015.

[16] Perge, J., *et al*., "Intra-day signal instabilities affect decoding performance in an intracortical neural interface system," *Journal of neural engineering,* vol. 10, no. 3, 2013.

[17] Chestek, C., Gilja, V., Nuyujukian, P., Foster, J., Fan, J., Kaufman, M., Churchland, M., Rivera-Alvidrez, Z., Cunningham, J., Ryu, S., and Shenoy, K., "Long-term stability of neural prosthetic control signals from silicon cortical arrays in rhesus macaque motor cortex," *Journal of Neural Engineering,* vol. 8, no. 4, pp. 1-21, 2011.

[18] Serruya, M., Hatsopoulos, N., Paninski, L., and Donoghue, J., "Instant neural control of a movement signal," *Nature,* vol. 416, no. 6877, pp. 141-142, 2002.

[19] Wu, W., Shaikhouni, A., Donoghue, J., and Black, M., "Closed-loop neural control of cursor motion using a kalman filter," *26th Annual International Conference of the IEEE Engineering in Medicine and Biology Society,* vol. 6, pp. 4126-4129, 2004.

[20] Wu, W., "Inferring hand motion from multi-cell recordings in motor cortex using a kalman filter," *Workshop on Motor Control in Humans and Robots: On the Interplay of Real Brains and Artifical Devices,* pp. 1-8, 2002.

[21] Kostov, A., Andrews, B., Popovic, D., Stein, R., and Armstrong, W., "Machine learning in control of functional electrical stimulation (fes) for locomotion," IEEE Trans. Biomed. Eng., vol. 42, pp. 541-551, 1995.

[22] Gage, G. J., "Co-adaptive kalman filtering in a naïve rat cortical control task," *IEEE Conference on Engineering in Medicine and Biology Conference,* vol. 6, pp. 4367-4370, 2004.

[23] Brockwell, E., Rojas, L., and Kass, R. E., "Recursive bayesian decoding of motor cortical signals by particle filtering," *Journal of neurophysiology,* vol. 91, no. 4, pp. 1899-1907, 2004.

[24] Van Staveren, G., Buitenweg, J., Heida, T., and Ruitten, W., "Wave shape classification of spontaneaous neural activity in cortical cultures on micro-electrode arrays," *Proceedings 24th Annual Conference of the EMBS/BMES society,* vol. 3, pp. 2010-2011, 2002.

[25] Kim, K., and Kim, S., "Neural spike sorting under nearly 0-db signal-to-noise ratio using nonlinear energy operator and artificial neural-network classifier," *IEEE Transactions on Biomedical Engineering,* vol. 47, pp. 1406-1411, 2000.

[26] Kim, K., and Kim, S., "Advantage of support vector machine for neural spike train decoding under spike sorting errors," *Proceedings of the 27th Engineering in Medicine and Biology Conference,* vol. 5, pp. 5280-5283, 2005.

[27] Bialek, W., "Theoretical physics meets experimental neurobiolog," *E. Jen (ed.) Lectures in Complex Systems, SFI Studies in the Science of Complexity,* vol. 2, pp. 413-595, 1989.

[28] Dorsey, J., "Continuous and discrete control system modelling, identification, design and implementation," *McGraw Hill,* 2002.

[29] Chen, Z., Haykin, S., Eggermont, J., and Becker, S., "Correlative learning: A basis for brain and adaptive systems, *John Wiley & Sons,* 2008.

[30] Hagan, M., De Jesus, O., and Schultz, R., "Training recurrent networks for filtering and control," L. Medsker, E. L.C. Jain (eds.) chap. 12 in *Recurrent Neural Networks: Design and Applications*, pp. 311-340, 1999.
[31] Narendra, K., and Mukhopadhyay, S., "Adaptive control using neural networks and approximate models," *IEEE Transactions on Neural Networks*, vol. 8, pp. 475-485, 1997.
[32] Lang, K., Waibel, A., and Hinton, G., "A time delay neural network architecture for isolated word recognition," *Neural Networks*, vol. 3, no. 1, pp. 23-44, 1990.
[33] Lin, T., Giles, C., Horne, B., and Kung, S., "A delay damage model selection algorithm for narx neural networks," *IEEE Transactions on Signal Processing*, vol. 45, no. 11, pp. 2719-2730, 1997.
[34] Hestenes, M., and Stiefel, E., "Methods of conjugate gradients for solving linear systems," *Journal of Research of the National Bureau of Standard*, vol. 49, no. 6, 1952.
[35] Møller, M., "Original contribution: A scaled conjugate gradient algorithm for fast supervised learning," *Neural Netw.*, vol. 6, no. 4, pp. 525-533, 1993.
[36] Wessberg, J., and Nicolelis, M.A., "Optimizing a linear algorithm for real-time robotic control using chronic cortical ensemble recordings in monkeys," *J. Cognitive Neuroscience*, vol. 16, no. 6, pp. 1022-1035, 2004.
[37] Wang, Y., Kim, S., and Principe, J., "Comparison of TDNN training algorithms in brain machine interfaces," *IEEE International Joint Conference on Neural Networks*, vol. 4, pp. 2459-2462, 2005.
[38] Yu, G., and Lin, Y., "A methodology for selecting subset autoregressive time series models," *Journal of Time Series Analysis*, vol. 12, no. 4, pp. 363-373, 1989.
[39] Freeman, J. A., and Skapura, D. M., "Neural networks-algorithms, applications, and programming techniques," *Addison–Wesley*, 1991.
[40] Bishop, M.," Neural Networks for Pattern Recognition," *Clarendon Press*, 1995.
[41] Sameni, R., Vrins, F., Parmentier, F., Herail, F., Vigneron, V., Verleysen, M., Jutten, C., and Shamsollahi, M., "Electrode selection for noninvasive fetal electrocardiogram extraction using mutual information criteria," *MaxEnt2006 proceedings-26th International Workshop on Bayesian Inference and Maximum Entropy Methods in Science and Engineering*, vol. 872, pp. 97-104, 2006.
[42] Guyon, I., and Elisseeff, A., "An introduction to variable and feature selection," *Journal of Machine Learning Research*, pp. 1157-1182, 2003.
[43] Shakil, M., Elshafei, M., Habib, and M., Maleki, F., "Soft sensor for nox and O2 using dynamic neural networks," *Computers and Electrical Engineering*, vol. 35, no. 4, pp. 578-586, 2009.
[44] Vrins, F., Lee, J., Verleysen, M., Vigneron, V., and Jutten, C., "Improving independent component analysis performances by variable selection," *NNSP'2003 proceedings-Neural Networks for Signal Processing*, pp. 359-368, 2003.
[45] Sharf, L. L., "Statistical Signal Processing," *Addison-Wesley Series in Electrical and Computer Engineering*, 1991.
[46] Vigneron, V., Syed, T., Barlovatz-Meimon, G., and Lelandais, S., "Adaptive filtering and hypothesis testing: Application to cancerous cells detection," *Pattern Recognition Letters*, vol. 31, no. 14, pp. 2214-2224, 2010.
[47] Tassi, P., and Lagait, S., "Théorie des probabilités en vue des applications statistiques," *Ecole nationale supérieure du pétrole et des moteurs*, 1990.
[48] Kifouche, A., Vigneron, V., Shamsollahi, M. B., and Guessoum, A., "Decoding hand trajectory from primary motor cortex ecog using time delay neural network," *Engineering Applications of Neural Networks-15th International Conference*, pp. 237-247, 2014.
[49] Dayen, P., and Abbott, L., "Theoretical Neuroscience computationa and mathematical modeling of neural systems," *The MIT Press*, 2001.