Neural Networks 137 (2021) 174–187

Contents lists available at ScienceDirect

Neural Networks

journal homepage: www.elsevier.com/locate/neunet

# Building an adaptive interface via unsupervised tracking of latent manifolds

Fabio Rizzoglio [a,b,c,*], Maura Casadio [a,1], Dalia De Santis [b,c,d,1], Ferdinando A. Mussa-Ivaldi [b,c,1]

[a] Department of Informatics, Bioengineering, Robotics and Systems Engineering, University of Genoa, 16145 Genoa, Italy
[b] Department of Physiology, Feinberg School of Medicine, Northwestern University, Chicago, IL, 60611, USA
[c] Shirley Ryan Ability Lab, Chicago, IL, 60611, USA
[d] Department of Robotics, Brain and Cognitive Sciences, Istituto Italiano di Tecnologia, Via Enrico Melen 83, 16152, Genoa, Italy

ARTICLE INFO

*Article history:*
Received 3 July 2020
Received in revised form 16 November 2020
Accepted 14 January 2021
Available online 20 January 2021

*Keywords:*
Autoencoder networks
Decoder adaptation
Human–machine interaction
Motor learning
Body–machine interface

ABSTRACT

In human–machine interfaces, decoder calibration is critical to enable an effective and seamless interaction with the machine. However, recalibration is often necessary as the decoder off-line predictive power does not generally imply ease-of-use, due to closed loop dynamics and user adaptation that cannot be accounted for during the calibration procedure. Here, we propose an adaptive interface that makes use of a non-linear autoencoder trained iteratively to perform online manifold identification and tracking, with the dual goal of reducing the need for interface recalibration and enhancing human–machine joint performance. Importantly, the proposed approach avoids interrupting the operation of the device and it neither relies on information about the state of the task, nor on the existence of a stable neural or movement manifold, allowing it to be applied in the earliest stages of interface operation, when the formation of new neural strategies is still on-going.

In order to more directly test the performance of our algorithm, we defined the autoencoder latent space as the control space of a body–machine interface. After an initial offline parameter tuning, we evaluated the performance of the adaptive interface versus that of a static decoder in approximating the evolving low-dimensional manifold of users simultaneously learning to perform reaching movements within the latent space. Results show that the adaptive approach increased the representational efficiency of the interface decoder. Concurrently, it significantly improved users' task-related performance, indicating that the development of a more accurate internal model is encouraged by the online co-adaptation process.



## 1. Introduction

Understanding how we learn to interact with a machine is of primary importance when designing human–machine interfaces. The ability of the nervous system to reorganize its structure and create novel neural pathways in response to learning is widely recognized (Dayan & Cohen, 2011; Donati et al., 2016; Kandel et al., 2000). Motor training and skill learning rely on neural plasticity in the nervous system enabling each individual to adapt to new environments (Shadmehr & Mussa-Ivaldi, 1994; Shadmehr et al., 2010; Wei et al., 2005), to learn new tasks (Mawase et al., 2017) and to efficiently operate a variety of devices (Danziger et al., 2009). On the other hand, to optimize the human interactions with devices, it is necessary to design interfaces that understand the user's abilities, preferences, and intentions.

The approach followed by brain–machine interfaces (BMIs) (Carmena et al., 2003; Shenoy & Carmena, 2014) is to decode intended actions from high dimensional neural recordings, and then convert (or encode) the decoded intentions into a lower dimensional set of commands for operating a device (*e.g.*, moving a computer cursor or robotic manipulator). Similarly, body–machine interfaces (BoMIs) extract a low dimensional control space from more down-stream information related to the execution of voluntary movements, as body kinematics and muscle activity (Casadio et al., 2012; Farshchiansadegh et al., 2014; Miehlbradt et al., 2018; Rizzoglio et al., 2020).

The efficiency of an interface in transferring the user's input to the device is tightly dependent on the joint user-decoder performance. There are at least two main ingredients that make the acquisition of interface control proficiency possible, (i) user learning and (ii) decoder adaptation.

* Corresponding author at: Department of Informatics, Bioengineering, Robotics and Systems Engineering, University of Genoa, 16145 Genoa, Italy.
*E-mail addresses:* fabio.rizzoglio@edu.unige.it (F. Rizzoglio), maura.casadio@unige.it (M. Casadio), dalia.desantis@iit.it (D. De Santis), sandro@northwestern.edu (F.A. Mussa-Ivaldi).

[1] These authors are joint senior authors on this work.

https://doi.org/10.1016/j.neunet.2021.01.009

In a typical BMI (Shanechi, 2017) the decoder is initialized by a supervised calibration procedure, which creates a map based on labelled examples of desired actions and corresponding neural activities. However, a good calibration is not sufficient to achieve proficient BMI control, as the performance of the decoder often degrades and fluctuates when evaluated online, due to imperfect predictions, recording instabilities, changes in neuronal properties, attentional changes, and changes brought about by learning (Barrese et al., 2013; Downey et al., 2018). BoMI decoders suffer from similar issues. The decoder of a BoMI, also referred to as "forward map", is initialized by unsupervised identification of the low-dimensional, latent manifold of unconstrained users' movements recorded during an initial calibration (Casadio et al., 2010). This latent manifold is expected to change with subsequent practice. Therefore, a discrepancy is likely to develop in time between the evolving latent manifold of the user and the initial BoMI forward map. Indeed, it has been observed that extensive practice with a BoMI led to the consolidation of task-specific movement strategies (Pierella et al., 2017) and several studies in brain–machine interfaces demonstrated the existence of a stable manifold of neural activity linked to BMI use and interpreted this as the result of neural adaptation following extensive practice with the interface (Gallego et al., 2020; Ganguly & Carmena, 2009; Oweiss & Badreldin, 2015; Shenoy & Carmena, 2014).

The consequence of adopting a fixed activity-intention map after its initial offline tuning, is that the user is left with the burden of learning how to use the interface for achieving new goals within new operating conditions that the decoder has not been optimized on.

Closed-loop supervised decoder adaptation in BMIs has been proposed as an effective way to increase decoder performance during use. This was the case when allowing the decoder parameters to smoothly change according to the inferred movement goal during within-session interface operation (Dangi et al., 2013; Orsborn et al., 2014). A very recent study considered instead the case of performance loss arising from decoder instabilities across sessions. They showed that by calibrating the decoder using features within the manifold of stable activity of the recorded neurons, performance of the BMI can be reliably recovered by manifold alignment across sessions (Degenhart et al., 2020). Despite contributing encouraging results, these current perspectives rely either on the knowledge of user intent or on the presence of an established manifold of neural activity linked to the use of the BMI. Hence, they cannot be easily extended to facilitate the use of the interface when the movement goal is unknown or when the formation of new neural strategies is still ongoing and the activity manifold has not yet consolidated (Oby et al., 2019).

Given the current limitations of closed-loop decoder adaptation, we propose a procedure for facilitating interface operation that does not rely on estimates of user intent and can be applied from the very initial stages of learning. The procedure is initially developed for application with body–machine interfaces and exploits a non-linear autoencoder (AE) network (Kramer, 1991) trained iteratively to identify and track the evolution of the latent manifold of its inputs.

Previous work from our group suggested that individuals training with a linear BoMI that was adapted iteratively based on movement statistics increased movement efficiency compared to a fixed interface and were able to develop a more faithful internal representation of the BoMI forward map (De Santis et al., 2018; De Santis & Mussa-Ivaldi, 2020). Building on these results, the hypothesis of this study is that the operation of the interface will be facilitated if the BoMI forward map is updated online so as to match the evolving manifold of the user's input. That is, in other words, if the output manifold of the forward map and the user's movement manifold were isomorphic.

The major advantages of the proposed approach are that it can be applied in the earliest stages of interface operation, it does not require interrupting the operation of the interface, and it does not need to incorporate information about the state of the task, in contrast with other adaptive interfaces described in the literature (DiGiovanna et al., 2009; Mahmoudi et al., 2008; Sanchez et al., 2009). Moreover, the use of a non-linear algorithm allows representing a curved manifold that is more consistent with the variance structure of input signals (Portnova-Fahreeva et al., 2020) compared to a linear approximation.

We decided to test our hypothesis within the framework of body–machine interfaces, as they allow evaluating learning of a low-dimensional manifold in a more immediate way, *i.e.* without requiring spike sorting or other procedures necessary for preprocessing neural activity. In addition, there has been only preliminary evidence regarding the applicability of non-linear AEs to the control of low dimensional devices (Pierella et al., 2018; Vujaklija et al., 2018). Hence, the results of this work will also serve as validation for the use of AEs in BoMI applications.

The following sections are structured as follows.

We first describe some preliminary concepts (Section 2. Preliminaries). We introduce the framework of the body–machine interface (Section 2.1) and how it can be implemented using non-linear AE (Section 2.2). At this stage, we assume the interface to be fixed after calibration (*i.e., fixed autoencoder).*

Then, we propose an implementation of iterative learning of a non-stationary latent manifold and we call this *adaptive autoencoder* (a-AE) (3.1). Finally, we give details of the two experiments we carried out for validating the performance of the a-AE both offline (3.3) and online (3.4). In the first study, we performed a sensitivity tuning to identify the hyperparameters that allow the a-AE to smoothly track a non-stationary movement distribution. This operation was conducted offline on real data collected from a naive BoMI user performing a reaching task with a fixed BoMI. The second study was conducted with the purpose of testing if training with an adaptive autoencoder facilitates user adaptation to the interface when compared to a fixed BoMI. In other words, we verified whether human–machine interaction might benefit if modelled as a bi-directional adaptation (*i.e., co-adaptation*) of a learning brain with a learning interface. During this online test, the a-AE hyperparameters were set as those that yielded the best performance in the first study.

## 2. Preliminaries

### 2.1. Body–machine interfaces

BoMIs (Casadio et al., 2012) are a class of human–machine interfaces that provide a link between human body motions and an external machine. The general purpose of a BoMI is to enable its user to retain a complete or shared control over the device through signals derived from the user's body. The BoMI exploits the assumption that the signals needed to control the external device lie on a low-dimensional manifold embedded within the higher-dimensional body signal space. In a BoMI, sensors (*e.g.*, infrared markers Casadio et al., 2010, inertial measurement units Farshchiansadegh et al., 2014, electromyographic electrodes Rizzoglio et al., 2020) are placed on the upper body of the user to record a $d$-dimensional vector of body signal, $q$. A forward map $f(.)$ is then constructed to transform $q$ into an $m$-dimensional control vector, $p$, (with $m < d$) encoding the commands for controlling an external device (*e.g.*, computer cursor, powered wheelchair, quadcopter):

$$p = f(q) \tag{1}$$

The BoMI forward map $f$ is determined in a calibration session, where BoMI users are asked to freely move their limbs so as to

explore their range of motion. This procedure is not task-related, as users do not receive visual feedback of their movements and are not yet connected to the device. Data collected during calibration are then used as training set for a dimensionality reduction (DR) algorithm that derives $f$ by extracting the low-dimensional, $m$-D, manifold in which the highest amount of variance of body signals is available. Common iterations of BoMIs exploit linear DR methods to extract such low-dimensional manifold, *e.g.* Principal Component Analysis (PCA) (Wold et al., 1987), or Kalman Filter (Seáñez González et al., 2016). In this study, we explore the use of a non-linear DR algorithm, autoencoder networks.

### 2.2. Fixed autoencoder

AEs are unsupervised artificial neural networks capable of learning efficient lower dimensional representations of the input data without labelled data. An AE is a cascade of two components: an encoder $\boldsymbol{E}$ that converts the inputs to a lower dimensional latent representation, or code, followed by a decoder $\boldsymbol{D}$ that converts the latent representation into the outputs, with the same dimensions as the inputs. An AE attempts at learning to perform the identity map, copying its input, $\boldsymbol{q}$, to its output, $\hat{\boldsymbol{q}}(\boldsymbol{w}, \boldsymbol{b})$, with minimum loss, thus learning an efficient internal representation of the inputs within their latent manifold. This is achieved by minimizing the mean squared error between $\boldsymbol{q}$ and $\hat{\boldsymbol{q}}$:

$$\boldsymbol{J}\left(\boldsymbol{q}, \hat{\boldsymbol{q}}\right) = \frac{1}{k}\sum_{i=1}^{k}\sqrt{\left(\hat{\boldsymbol{q}}_{\boldsymbol{i}} - \boldsymbol{q}_{\boldsymbol{i}}\right)^2} \tag{2}$$

with $\boldsymbol{k}$ being the number of samples of $\boldsymbol{q}$. $\boldsymbol{w}$ and $\boldsymbol{b}$ are the parameters of the AE, representing the matrices of weights and biases of each layer of the network, respectively. The update rule for the AE parameters is:

$$\boldsymbol{w} \leftarrow \boldsymbol{w} - \alpha\nabla_{w}\boldsymbol{J}, \quad \boldsymbol{b} \leftarrow \boldsymbol{b} - \alpha\nabla_{b}\boldsymbol{J} \tag{3}$$

In this study, we used an AE with five layers (see Fig. 1) and Batch Gradient Descent (BGD) (Ruder, 2016) to minimize $\boldsymbol{J}$. As in the case of the AE proposed by Kramer (1991), we chose a non-linear activation function for the hidden layers and a linear activation function for the code and output layers. We did not opt for a deeper network because we aimed at employing a parsimonious architecture. An AE fed with the n-dimensional vector containing the body-signals ($\boldsymbol{q}$) applies the following transformations:

$$\boldsymbol{p} = \boldsymbol{E}\left(\boldsymbol{q}\right) = \boldsymbol{w}_2\left(\tanh\left(\boldsymbol{w}_1\boldsymbol{q} + \boldsymbol{b}_1\right)\right) + \boldsymbol{b}_2 \tag{4}$$

$$\hat{\boldsymbol{q}} = \boldsymbol{D}\left(\boldsymbol{p}\right) = \boldsymbol{w}_4\left(\tanh\left(\boldsymbol{w}_3\boldsymbol{p} + \boldsymbol{b}_3\right)\right) + \boldsymbol{b}_4 \tag{5}$$

where $\boldsymbol{w_i}$ and $\boldsymbol{b_i}$, $i = 1{:}4$ are the matrices of weights and biases of the $i$th layer, while $\hat{\boldsymbol{q}}$ is the n-dimensional vector containing the body-signals reconstructed by the AE.

In the BoMI context, the encoder sub-network $\boldsymbol{E}$ can be used as a forward map in Eq. (1). Thus, the vector $\boldsymbol{p}$ includes the commands for the control of the m-dimensional external device. Importantly, unlike other non-linear methods for DR (*e.g.*, Isomap Tenenbaum et al., 2000), after training the AE on a data set, the encoder $\boldsymbol{E}$ projects new incoming data on the same latent manifold derived from the training data.

## 3. Methods

### 3.1. Adaptive autoencoder

Here we describe the algorithm to adapt the interface accordingly to the evolving manifold of the user's input. We refer to this algorithm as *adaptive autoencoder* (a-AE).

The a-AE consists of a memory element and a learning element (Fig. 2). The memory element stores samples of the data depicting the state of the user (*e.g.*, body motion). The learning element intervenes with a certain frequency to update the parameters of the AE by maximizing its fitness to the data stored in the memory. The parameters that govern the response of the a-AE are:

- $f_u$, the frequency of learning element updates. An update iteration $n$ is performed every $1/f_u$ seconds.
- $k$, the size of the memory element, or equivalently the length of the training batch available at each update iteration n - $\boldsymbol{q}^{(n)}$, the *history batch*.
- $\beta$, the number of BGD steps over the history batch $\boldsymbol{q}^{(n)}$. Increasing $\beta$ increases the goodness of fit of the a-AE on the history batch.
- $\alpha$, the learning rate parameter, which determines the size of each step taken in the direction of the gradient of $J$ (Eq. (3)). Higher values of $\alpha$ accelerate convergence towards the minimum of $J$.

Every $1/f_u$ seconds, the learning element triggers an update iteration $n$ to adapt the AE parameters to the history batch stored in the memory. Namely, it runs $\beta$ BGD steps over $\boldsymbol{q}^{(n)}$ as in Eq. (3). To facilitate the a-AE tracking of the user's latent manifold, the AE parameters are updated via transfer learning (Bengio, 2012). Namely, at the start of each update iteration, the parameters of the AE are set to be equal to those of the previous iteration. In the generic $n$ update iteration, the history batch $\boldsymbol{q}^{(n)}$ consists of $k$ samples. If $k \geq f_s/f_u$, where $f_s$ is the sampling frequency of $q$, then $k$–$(f_s/f_u)$ are samples from the preceding update step and $(f_s/f_u)$ are new samples. Instead, if $k < f_s/f_u$ there will be no overlap between successive training batches. The pseudocode of the a-AE is shown in Table 1. The block diagram of the a-AE that we tested online is shown in Fig. 2.

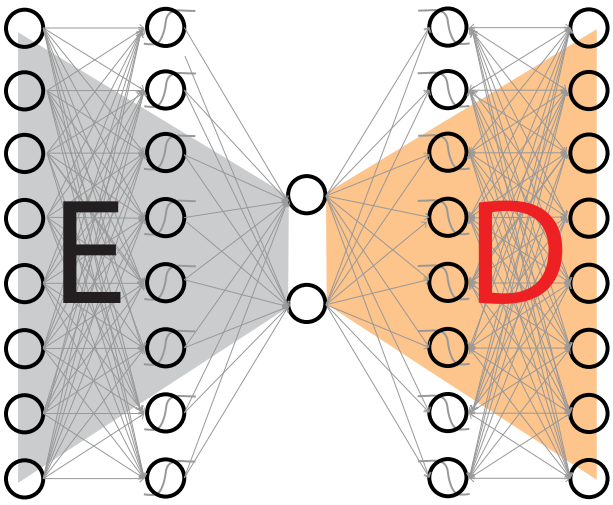



**Fig. 1.** Autoencoder (AE) network structure used in this study. The encoder part was a densely connected network that transformed an eight-dimensional input layer into a first hidden layer of the same dimensionality and subsequently a two-dimensional code layer. The decoder transformed the 2D latent space into a second 8D hidden layer and finally back to an 8D output layer. The first and second hidden layers applied a non-linear transformation (*i.e.*, hyperbolic tangent) to their inputs.

### 3.2. Experimental methods

This study consisted of two experiments. The first was an offline test of the proposed adaptive algorithm. The results of this experiment were used to evaluate the adaptive algorithm tracking performance and its sensitivity over the choice of the learning parameters. In the second experiment, we tested the adaptive

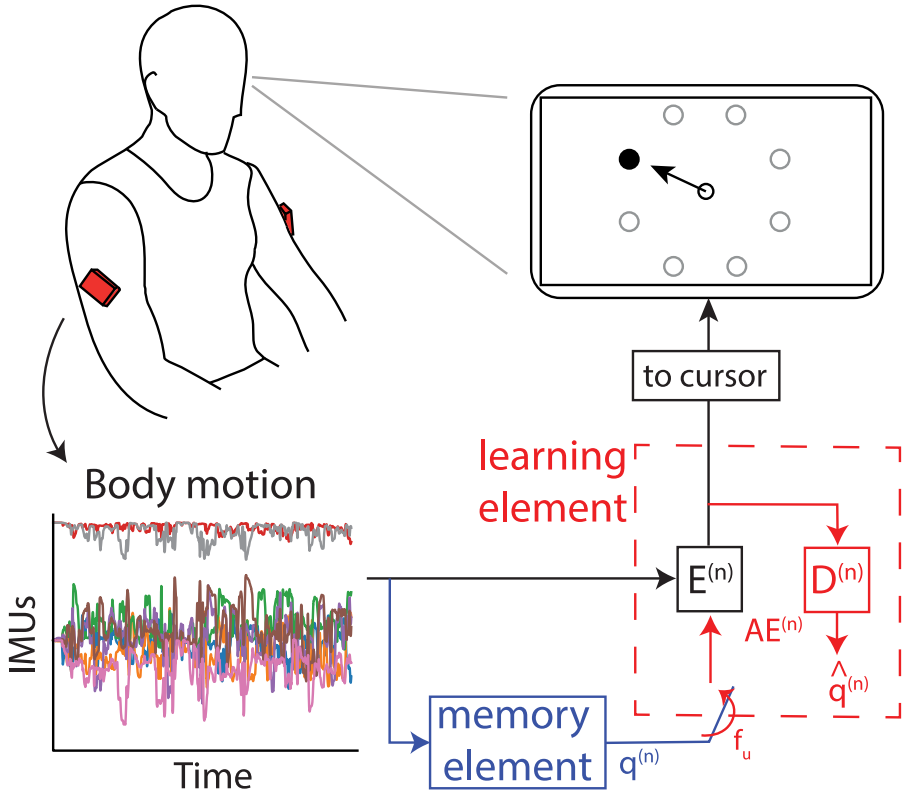



**Fig. 2.** Block diagram of the adaptive autoencoder. Body motion recoded by IMUs, shown as orange boxes, is fed to the encoder $\boldsymbol{E}^{(n)}$ and the memory element (blue). $\boldsymbol{E}^{(n)}$ extracts the user's latent space, which is in turn directly converted into coordinates of the computer cursor. Every $1/f_u$ seconds the learning element (red) activates, and the history batch $\boldsymbol{q}^{(n)}$ stored in the memory element is fed through $\boldsymbol{E}^{(n)}$ and $\boldsymbol{D}^{(n)}$ to obtain $\hat{\boldsymbol{q}}^{(n)}$. The parameters of $AE^{(n)}$ are finally updated via minimization of the mean squared error between $\hat{\boldsymbol{q}}^{(n)}$ and $\boldsymbol{q}^{(n)}$.

algorithm by closing the loop of the BoMI. Specifically, we compared static and co-adaptive non-linear BoMIs when controlling the two coordinates of a computer cursor using body signals derived from motions of the upper limbs with an embedding dimensionality of six.

We recorded the motion of the arms with two inertial measurement units (IMUs) (3 Space Sensors, *Yost Labs, Portsmouth, OH, USA*). The sensors were placed bilaterally as shown in Fig. 2. The IMU sensors used in this study derive orientation in the quaternion format using a complementary filter that integrates raw measurements from accelerometer, gyroscope, and magnetometer. However, we excluded the magnetometer from the IMU complementary filter due to the presence of ferromagnetic structures and transient disturbances that made its measurements unreliable and noisy. We therefore recorded an eight-dimensional movement signal. The BoMI software was custom coded in C#.

### 3.3. Study I: Offline sensitivity tuning

The sensitivity tuning of the a-AE was carried out as an offline simulation using the movement data recorded from a naive individual performing reaching movements with a BoMI. The procedure aimed at selecting the hyperparameter values that would allow the adaptive AE to attain three conditions: (1) providing a stable representation of the user's movement statistics, *i.e.* by converging towards a minimum $J$ over time in Eq. (2); (2) minimizing the discontinuity of the interface at the moment of the encoder update; (3) changing at a rate that is neither too slow nor too fast. The discontinuity of the interface is a critical point because it might affect its usability online when operating an external device. For example, in this study, a discontinuous adaptive interface would produce a sudden jump of the computer cursor, which could be perceived as a perturbation and possibly disrupt the user's experience with the BoMI.

**Table 1**
Pseudo-code for the a-AE.

**Input**: $k$, $\alpha$, $\beta$, $f_u$, $n = 1$, $t_{elapsed} = 0$
*# Initialize* $AE^{(0)} = \{E^{(0)}, D^{(0)}\}$ *with calibration data set* $\boldsymbol{q}^{(0)}$
$AE^{(0)} = f(\boldsymbol{w}^{(0)}, \boldsymbol{b}^{(0)})$, $\boldsymbol{q}^{(1)} = \boldsymbol{q}^{(0)}$
**while** not terminated **do**:
  *# 1. Memory element: update the history batch* $\boldsymbol{q}^{(n)}$:
  $\boldsymbol{q}^{(n)} = \text{append}(\boldsymbol{q}^{(n)}, q_i)$
  **if** size($\boldsymbol{q}^{(n)}$> $k$), pop($\boldsymbol{q}^{(n)}$[0])
  *# 2. activate the Learning element*
    **if** $t_{elapsed} = 1 \backslash f_u$
      *# Step I. Transfer learning on* $\boldsymbol{w}$ *and* $\boldsymbol{b}$
      $\boldsymbol{w}^{(n)} \leftarrow \boldsymbol{w}^{(n-1)}$, $\boldsymbol{b}^{(n)} \leftarrow \boldsymbol{b}^{(n-1)}$
      $AE^{(n)} = f(\boldsymbol{w}^{(n)}, \boldsymbol{b}^{(n)})$
      *# Step II. Run BGD on* $\boldsymbol{w}$ *and* $\boldsymbol{b}$
      **for** step in $\beta$
      $\hat{\boldsymbol{q}}^{(n)} = D^{(n)}(E^{(n)}(\boldsymbol{q}^{(n)}))$
      $\boldsymbol{w}^{(n)} \leftarrow \boldsymbol{w}^{(n)} - \alpha \nabla_{\boldsymbol{w}} \boldsymbol{J}(\boldsymbol{q}^{(n)}, \hat{\boldsymbol{q}}^{(n)})$
      $\boldsymbol{b}^{(n)} \leftarrow \boldsymbol{b}^{(n)} - \alpha \nabla_{\boldsymbol{b}} \boldsymbol{J}(\boldsymbol{q}^{(n)}, \hat{\boldsymbol{q}}^{(n)})$
      $AE^{(n)} = f(\boldsymbol{w}^{(n)}, \boldsymbol{b}^{(n)})$
      **end**
      $\boldsymbol{q}^{(n+1)} = \boldsymbol{q}^{(n)}$
      $n = n + 1$
      $t_{elapsed} = 0$
    **end**
**end**
**Output**: the AE model at current iteration $n$, $AE^{(n)}$

Previous studies suggested that the learning curves of users operating a BoMI during a reaching task follow an exponential trend with time constants ranging from approximately 2 to 10 min (De Santis & Mussa-Ivaldi, 2020). This means that users

display a 50% improvement in control performance only after the first 2 min of practice. Hence, it is desirable for the interface to reach a set point within approximately 2 min when updated online. Therefore, we set this to be the target value for the network to converge (<1% residual error) during the offline sensitivity tuning. Note that this procedure involved the concurrent tuning of two hyperparameters, the size $k$ of the memory element, and the network learning rate $\alpha$. In order to estimate which values yielded the most ideal behaviour, we fit a double exponential function to the learning curve derived from each combination of hyperparameters. We used the rate of convergence, defined as the time constant (tau) of the fastest exponential as metric to select the best candidates for the online test according to the rule 5*tau∼120 s.

To find a batch size that was representative of BoMI user's movement dynamic, we tested three values of $k$ referring to short-term (2 s), mid-term (20 s) and long-term (60 s) memory. Previous studies with movement-based body–machine interfaces (Abdollahi et al., 2017; De Santis et al., 2018; De Santis & Mussa-Ivaldi, 2020; Pierella et al., 2017) have found that naïve users progress from several tens of seconds to complete a reaching movement towards a target to only a few seconds after some practice. Hence, in the context of a short bout (∼15–20 min) of reaching task practice with the interface, the three levels of memory provide each update step with the information regarding within trial variation (2 s), data distribution when reaching towards a few (1–5) different targets (20 s), and information regarding movements distributed over a greater region of space (60 s).

Then, we chose to test values of learning rate with three distinct orders of magnitude to evaluate which time scale would have allowed the adaptive AE to satisfy the 120 s criterion.

In this study, we set $\beta$ equal to 10 steps of BGD and the a-AE update rate ($1/f_u$) to 2 s. The choice of updating the AE parameters every 2s was made to ensure a continuous (and quick) update, that could keep track of the evolving movement manifold of the BoMI user online. Ideally, the update should have been performed as fast as possible. However, we had to consider the time that the preprocessing operations (*e.g.*, filling the memory buffer with the latest kinematic samples recorded) and the training of the AE would have required. Since all these operations took, on average, little less than 1s to be completed, we decided to put some cushion and fix the update rate to 2s, in order to maintain a constant update rate throughout the experiment.

We included the Adaptive Moment Estimation (Kingma & Ba, 2015) optimization to allow meeting this time requirements for online computation of the AE parameters. The a-AE ran in a custom-coded Python thread and all the optimization policies were implemented using Tensorflow (Abadi et al., 2016).

### 3.4. Study II: Online test of the adaptive autoencoder

In the second study we tested the adaptive autoencoder by closing the loop of the BoMI. During this online test, the a-AE hyperparameters were set as those that yielded the best performance in the first study. In our BoMI, the encoder $\boldsymbol{E}$ mapped the eight-dimensional IMU-vector ($\boldsymbol{q}$) into the x–y cursor vector $\boldsymbol{p}$:

$$\boldsymbol{p} = \boldsymbol{E}(\boldsymbol{q}) + \boldsymbol{p}_0 \tag{6}$$

The offset vector $\boldsymbol{p}_0$ was chosen to make a desired neutral position (*i.e.*, rest position) of the body-space match a corresponding reference position of the cursor. Moreover, the resulting workspace was then rotated and stretched in order to ensure complete coverage of the entire workspace of the cursor (Casadio et al., 2010). While the overall philosophy of the BoMI is to base its operation on a customized mapping, adapted to each individual user, here we had to sacrifice some of this philosophy to the need of obtaining data that could be compared across subjects. This required having all participants start from a common initial condition, *i.e.* from the same initial mapping. To attain a compromise between uniformity and customization, we decided to base the initial BoMI mapping on movement data from a single "representative" user. A single individual (age 25, male), who did not participate in the rest of the study, was asked to freely move the arms exploring the full range of motion for 60 s. Movement data, recorded during this calibration phase, were used as a training data set. The encoder derived from this training data set was taken as the initial encoder ($\boldsymbol{E}^{(0)}$). This same initial map was set for all participants.

#### 3.4.1. Participants

We enrolled twenty unimpaired subjects. They did not have any known history of neuromotor or musculoskeletal disorders and exhibited typical joint range of motion and muscle strength. All participants gave their signed informed consent prior to the test. All procedures were carried out in accordance with the ethical standards of the Declaration of Helsinki and Northwestern University IRB approved all human involvement in the study (IRB protocol #STU00057856). Participants were divided in two groups and assigned to one of the following study conditions:

- *Fixed map* (group **F**, N = 10, age 26.7 ± 6.9, four females): participants were assigned a constant encoder $\boldsymbol{E}^{(0)}$, defined by the common calibration procedure. Therefore, each individual was presented with the same map which did not change throughout the session.
- *Adaptive map* (group **A**, N = 10, age 26.3 ± 5.9, six females): in this group, the encoder adapted to participants' movements following the a-AE algorithm validated in the first experiment. Initially, the encoder was defined by the same common calibration procedure as in the previous group $\boldsymbol{E}^{(0)}$. Then, the adaptive procedure would continuously update the body-cursor mapping $E^t$ as a function of the evolving user's movements.

As a result of the tuning, the a-AE updated the network using a history-batch of the last 60 s of the IMU data. 10 steps of BGD with a learning rate $\alpha = 10^{-4}$ were applied to the network trained in the previous iteration.

#### 3.4.2. Protocol

The protocol consisted of a reaching task. The participant sat in front of a 24" LCD computer screen, positioned about 1 m away at eye level. The current position of the cursor and the targets was displayed on the screen as circles of 0.8 cm and 2.4 cm diameter respectively, similarly to previous studies involving a 2D reaching task (Farshchiansadegh et al., 2014). Participants were asked to move the cursor over the targets as rapidly as possible. The sequence of target presentations was the same for all the participants. A target was considered reached after the cursor had been kept inside the target for at least 250 ms. The chosen time allowed discriminating between movements actually directed to and ending in the target and movements that brushed by the target. The protocol, as summarized in Fig. 3, included two phases of reaching: training and test.

#### 3.4.3. Training phase

The training phase was divided in 8 epochs. Within an epoch, the participants had to reach 4 targets (Fig. 3, grey dots) 8 times each. After completing a reaching movement, participants had to move the cursor towards the next target of the training sequence. The order of target presentation was pseudorandomized, with the condition that each target was not presented again before

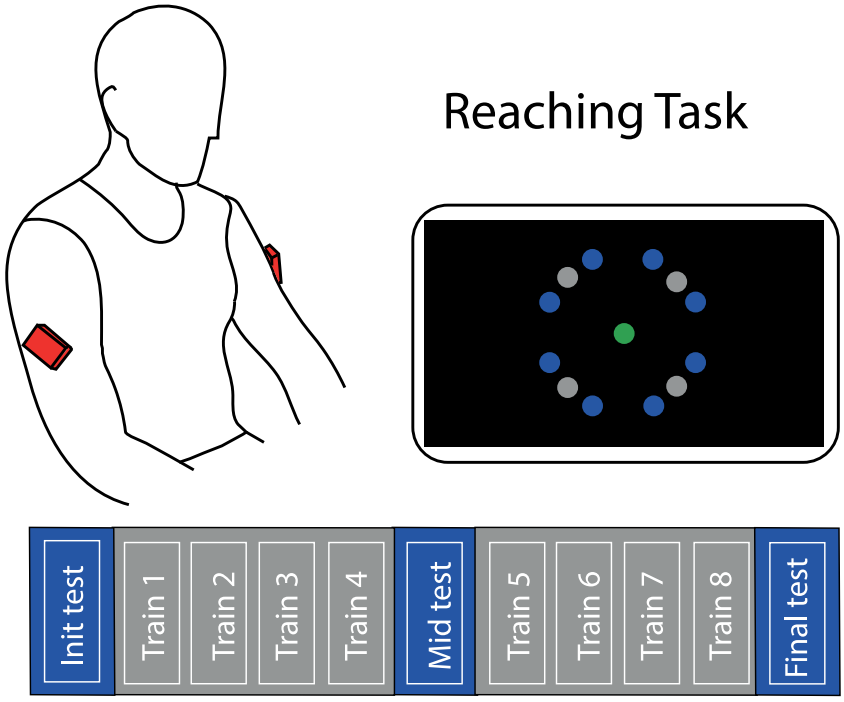



**Fig. 3.** Setup for the reaching task and training protocol. The participant was sitting in front of a computer monitor and was controlling a cursor using signals generated by IMUs (red boxes). Training (grey) and test (blue) targets were uniformly distributed on a circle. Four training targets were placed in four directions (45°+k90°), while eight test targets were placed in eight directions (22.5°+k45°). Each target was placed at the same distance L of 10.5 cm from HOME target.

all 4 targets had been reached. In the last repetition of the 4 targets visual feedback of the cursor was removed, and the participants were asked to stop moving when they believed to be in the target (blind trials). The goal of these blind trials was to establish if the participants were guided by error feedback or if, instead, they formed a feedforward command based on an internal representation of the cursor space.

During the first training epoch, the encoder for the adaptive group (A) was initialized with $\boldsymbol{E}^{(0)}$ and kept constant for 60 s (baseline). After that, the map was iteratively updated as described in Table 1. Update of the map was suspended during blind trials.

#### 3.4.4. Test phase

The participants practised a centre-out reaching task to eight target locations uniformly distributed on a circle (Fig. 3, blue targets). After each successful reaching, they were asked to move the cursor back to the central HOME target (Fig. 3, green target). In each test the eight targets were presented once.

A total of three tests were presented: an initial test as baseline before starting the experiment, a midway test after four epochs of training, and a final test at the end of the experiment. For the whole duration of each test, the control group (F) practised with $\boldsymbol{E}^{(0)}$, while the map update was suspended for the adaptive group (A). Hence both groups practised with the same map during the initial test epoch.

### 3.5. Outcome measures

#### 3.5.1. Sensitivity tuning

To evaluate the convergence and the stability of the a-AE in the offline sensitivity tuning we computed three metrics.

First, we quantified the tracking performance of the a-AE based on the **reconstruction error (RE).** RE is defined as the loss function described in Eq. (2), that was minimized every two seconds during the online retraining of the AE. A reduction of the reconstruction error implies that the mapping was providing a stable representation of the AE training data set.

To quantify the stability of the interface when using the a-AE, we considered the actual cursor vector $p^{(n)}$, and the cursor vector, $\hat{p}^{(n)}$, that would have been obtained without updating the encoder:

$$\begin{aligned} \hat{p}^{(n)} &= E^{(n-1)}(q^{(n)}) \\ p^{(n)} &= E^{(n)}(q^{(n)}) \end{aligned} \tag{7}$$

where $E^{(n-1)}$ and $E^{(n)}$ are the two time-consecutive encoders and $q^{(n)}$ is the movement set used in the training of the AE at the current update iteration. Note that, even though the a-AE at consecutive update iterations ($E^{(n-1)}$ and $E^{(n)}$) was trained with different movement set $q$, here we computed the cursor trajectory using the same set $q^{(n)}$. This allowed us to evaluate the **static jump**, defined as the $L_2$ norm between the final $k$ value of $\hat{p}^{(n)}$ (end of an update iteration) and the initial value of $p^{(n)}$ (beginning of the next update iteration):

$$static\ jump = \sqrt{\left(\hat{p}_k^{(n)} - p_1^{(n)}\right)^2} \tag{8}$$

Finally, we defined the **rate of change** between consecutive updates by computing the $L_2$ norm between $\hat{p}^{(n)}$ and $p^{(n)}$:

$$rate\ of\ change = \sqrt{\sum_{i=1}^{k}\left(\hat{p}_i^{(n)} - p_i^{(n)}\right)^2} \tag{9}$$

#### 3.5.2. Online test

Four different measures were chosen to evaluate and compare the performance of the fixed and the adaptive group during the online test of the a-AE. Two of these were explicitly associated with the task requirements:

- **Trials completed (TC)** number of reaching trials completed during the training phase within a specific time frame. We

computed the number of trials completed during baseline and then during time intervals of two minutes, including the final two minutes of training. Blind trials were not considered in this metric.

- **Endpoint Error (EE):** the Euclidean distance between the target position and the cursor position at the time of the cursor re-appearance during blind trials.

The other indicators measured performance not explicitly associated with the task requirements:

- **Linearity Index (LI):** maximum lateral deviation from the straight line connecting the beginning and end of cursor movement divided by distance between the same points. This is an index of straightness of cursor movements.
- **Movement Smoothness (MS):** number of peaks in the cursor velocity profile. We considered every peak larger than a threshold that was set to be 15% of the maximum speed of each trajectory. This is a measure of trajectory smoothness.

Besides analysing the task-related performances, we also wanted to examine how the parameters of the a-AE evolved in time as well as to compare the strategies of the fixed and the adaptive group. Three metrics were devised for such purposes.

To assess the tracking ability of the a-AE, we monitored the a-AE reconstruction error at each update iteration of online adaptation of the interface, as we did in the offline tuning. From this measure, we derived the **Variance Accounted For (VAF)**, defined as:

$$VAF = \left(1 - \frac{var\left(q - \hat{q}\right)}{var\left(q\right)}\right) * 100 \tag{10}$$

Higher values of VAF were associated with an increased goodness of fit of the AE at the $n$ update iteration, and consequently a sign of an enhanced tracking ability of the a-AE towards participant's latest movements. We then compared the VAF of the adaptive and the fixed group. In case of the a-AE successfully tracking user's movement manifold, we expected its VAF to be consistently higher than that of the fixed algorithm.

Both groups were presented with a non-linear encoder (Eq. (4)). We made the hypothesis that participants learned to minimize the "wasted" motion in the articulation space of the $q$'s. To characterize how the control strategy (or, equivalently, the movement distribution) was changing throughout training, we computed the **bi-dimensionality index**. It was defined as the VAF by an AE model trained over each of the eight training epochs. The AE architecture was the same as in Fig. 1.

Finally, we wanted to determine whether the AE encoders in the a-AE converged to a specific structure. This required evaluating the similarity between different AE structures. The problem of assessing the similarity between neural networks had been widely studied (Kornblith et al., 2019; Morcos et al., 2018). Here, we followed a procedure described in Raghu et al. (2017) that makes use of the Canonical Correlation Analysis (CCA) (Thompson, 2005). Specifically, we used the CCA to compare the final encoder structure obtained at the end of the training between any couple of participants $(i, j)$. The inputs of the CCA were:

$$\begin{aligned} \mathrm{p}_{\mathrm{i}}^{\mathrm{t_{end}}} &= \mathrm{E}_{\mathrm{i}}^{\mathrm{t_{end}}}(\mathrm{x}) \\ \mathrm{p}_{\mathrm{j}}^{\mathrm{t_{end}}} &= \mathrm{E}_{\mathrm{j}}^{\mathrm{t_{end}}}(\mathrm{x}) \end{aligned} \tag{11}$$

$x$ was designed as a vector of 3000 samples drawn from an 8D gaussian distribution. To resemble the distribution of a typical participant, the mean and covariance of $x$ were set equal to those of the movement data from the same participant we used during the offline tuning. Note that, by using the same synthetic input $x$, we ensured this metric to depend only on the structure of the encoder $E$. Since the CCA works by maximizing the correlation $\rho^m$ between its $m$ inputs, we determined the **representational similarity** as:

$$\text{similarity} = \frac{1}{2}\sum_{\mathrm{m}=1}^{2}\rho^{\mathrm{m}} \tag{12}$$

By maximizing the correlation between its inputs, the CCA ensures the representational similarity metric to be invariant to any affine transformation. Higher values of similarity implied that participants converged towards a unique encoder. As an example of the effect of the similarity between final AE structures in terms of cursor control, we applied to the IMU data recorded by a participant during the final test epoch the final encoders of two other participants, one with a high and one with a low similarity index respectively (the resulting cursor trajectories are shown in the result section).

### 3.6. Statistical analysis

To test whether the adaptive control affected the number of trials completed per unit time with respect to the fixed control, we performed a repeated measures analysis of variance (rANOVA) with time (level 1: baseline, level 2: last 2 min of training epoch) as within-subjects factor and group (level 1: adaptive, level 2: fixed) as between-subjects factor. We were interested in testing the group per time interaction, and the effect of the group. We did not test for the effect of time alone, as the duration of level 1 and level 2 was different.

Similarly, to test the effect of time and group on the other indicators related to kinematic performance during training we ran a rANOVA with time (1-2: first and eighth training epochs) as within-subjects factor and group (1-2: adaptive, fixed) as between-subjects factor.

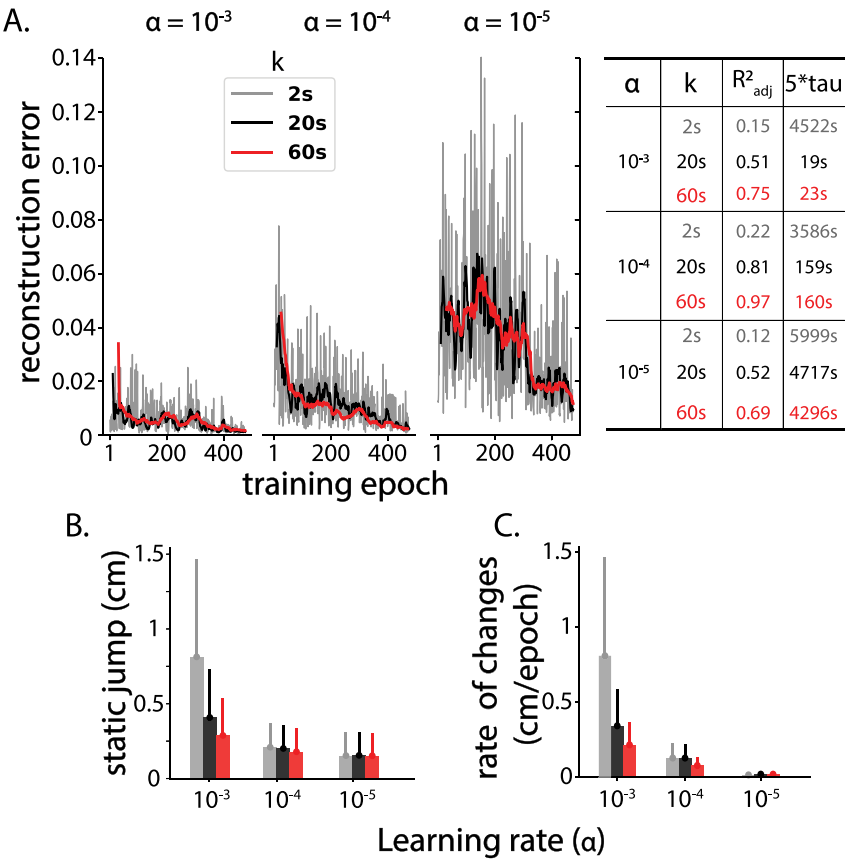
We verified that the assumptions of rANOVA were met by testing the sphericity of the data with the Mauchly's test and the normality of the data with the Anderson–Darling test. As a result, all data were normally distributed and spherical. Post-hoc analysis (Bonferroni correction) was carried out to verify statistically significant differences among factors whose effect was found to be significant. The threshold for significance was set at 0.05. For the post-hoc analysis, the Bonferroni-corrected threshold of significance was set to 0.025 (0.05/2). All analyses were performed in Statistica (Statsoft, Tulsa, OK, USA).

## 4. Results

### 4.1. Offline sensitivity tuning of the adaptive AE

Fig. 4A shows the reconstruction error during the a-AE sensitivity tuning for different values of learning rate $\alpha$ and memory batch size $k$, together with the $R^2_{adj}$ and the set point of convergence ($5^*tau$) derived from the exponential fitting. The smallest value of the learning rate ($\alpha = 10^{-5}$) prevented the algorithm to converge within an acceptable time-frame, as the set point of convergence was consistently above 4000s for every size of $k$. Using a short-time memory batch increased uncertainty in the values of the reconstruction error over the training epochs (Fig. 4A, grey lines). As a result, the exponential fitting yielded the lowest $R^2_{adj}$ for all the values of $\alpha$ tested. With a bigger memory batch and a learning rate $\alpha = 10^{-3}$, the adaptive algorithm fit user's movements in the fastest time frame (19s and 23s with a mid-term and a long-term history batch respectively). The learning rate of $10^{-4}$ had intermediate performance, allowing a significantly faster convergence than that of $10^{-5}$, and slightly slower than $10^{-3}$.

Fig. 4B shows the amplitude of cursor position discontinuity at the instant of the AE parameters' update. The higher the learning




| α | k | $R^2_{adj}$ | 5*tau |
|---|---|---|---|
| $10^{-3}$ | 2s | 0.15 | 4522s |
| | 20s | 0.51 | 19s |
| | 60s | 0.75 | 23s |
| $10^{-4}$ | 2s | 0.22 | 3586s |
| | 20s | 0.81 | 159s |
| | 60s | 0.97 | 160s |
| $10^{-5}$ | 2s | 0.12 | 5999s |
| | 20s | 0.52 | 4717s |
| | 60s | 0.69 | 4296s |

**Fig. 4.** Summary of sensitivity tuning results. Panel A: reconstruction values with $\alpha = 10^{-3}, 10^{-4}, 10^{-5}$ and three different values of history-batch size (2s — grey, 20s — black and 60s — red). Note that, on the *x*-axis, one training epoch is equal to 2s. A table containing the adjusted coefficient of determination $R^2_{adj}$ and the set point of convergence 5**tau* after fitting a double exponential is shown for each combination of hyperparameters. Panel B: mean and standard error of static jump values during training epochs. Panel C: mean and standard error representing the a-AE rate of change during training epochs. The same colour scheme was used in every panel.

rate, the bigger was the jump of the cursor. Vice versa, the bigger the size of the history batch, the smaller was the jump. A similar trend was found with the rate of changes of the AE map (Fig. 4C). Namely, the encoder had larger changes with larger learning rates and smaller history-batch sizes.

### 4.2. Online test of the adaptive AE

The adaptive group completed a significantly higher number of trials per unit time than the fixed group (rANOVA group effect: $F(1, 18) = 6.35, p = 0.02$, Fig. 5A). Specifically, we found a significant interaction between group and time (rANOVA group x time effect: $F(4, 72) = 5.5, p = 0.03$). At the beginning of the training, in the first 60s when the map was equal for all participants (baseline), both the adaptive and the fixed group completed the same number of trials (post-hoc comparison baseline: $p = 1$). However, immediately after, the group that started working with the adaptive AE increased the rate of trials completed, with respect to the group that continued working with the initial map. Performance differences increased over time and were found to be significant in the final two minutes of training (post-hoc comparison last 2min: $p = 0.009$).

Both groups improved their accuracy over training time (rANOVA time effect: $F(1, 18) = 24.6, p < 0.001$). With the adaptive algorithm, participants outperformed the fixed group in terms of accuracy when reaching the targets during the blind trials (Fig. 5B). However, this difference was not found statistically significant (rANOVA group effect: $F(1, 18) = 2.5, p = 0.13$).

Finally, with practice all participants moved the cursor along straighter lines (rANOVA time effect:$F(1, 18) = 70.97, p < 0.001$, Fig. 5C) and with increasing smoothness (rANOVA time effect: $F(1, 18) = 116.8, p < 0.001$), Fig. 5D). At the beginning of the training phase, the initial performance of the fixed and the adaptive group was comparable for each metric. Performance differences became more evident during the second epoch of training, even though not significantly.

Participants from both groups learned to organize their movements in a latent manifold that progressively evolved towards a bi-dimensional structure across training (Fig. 6A). In fact, the bi-dimensionality index increased with training for both groups and eventually coincided at the end of the training.

Noticeably, we recorded a significant difference when looking at the Variance Accounted For by the AE map over the training duration. The fixed map (Fig. 6B, black lines) was not able to capture the movement manifold of its users, as the values of VAF plateaued at around 25% for the entire duration of the reaching task. On the other hand, allowing the AE map to change over time consistently increased the fitness of the a-AE to the user's movement manifold (Fig. 6B, red lines) and eventually reached the peak of 80% VAF at the end of the training duration.

Fig. 7 shows the similarity matrix representing the similarity between the encoders obtained at the end of the training phase for the adaptive group. Participants were ranked according to the inter-subject similarity between final AE structure. Six participants converged towards fairly comparable encoders (S1, S2, S4, S6, S8, S10 - 6 × 6 matrix in the top left corner, Fig. 7), while four did not (S3, S5, S7, S9).

An example of how such similarity might have impacted the control of the cursor in shown in Fig. 8. Namely, if we were to

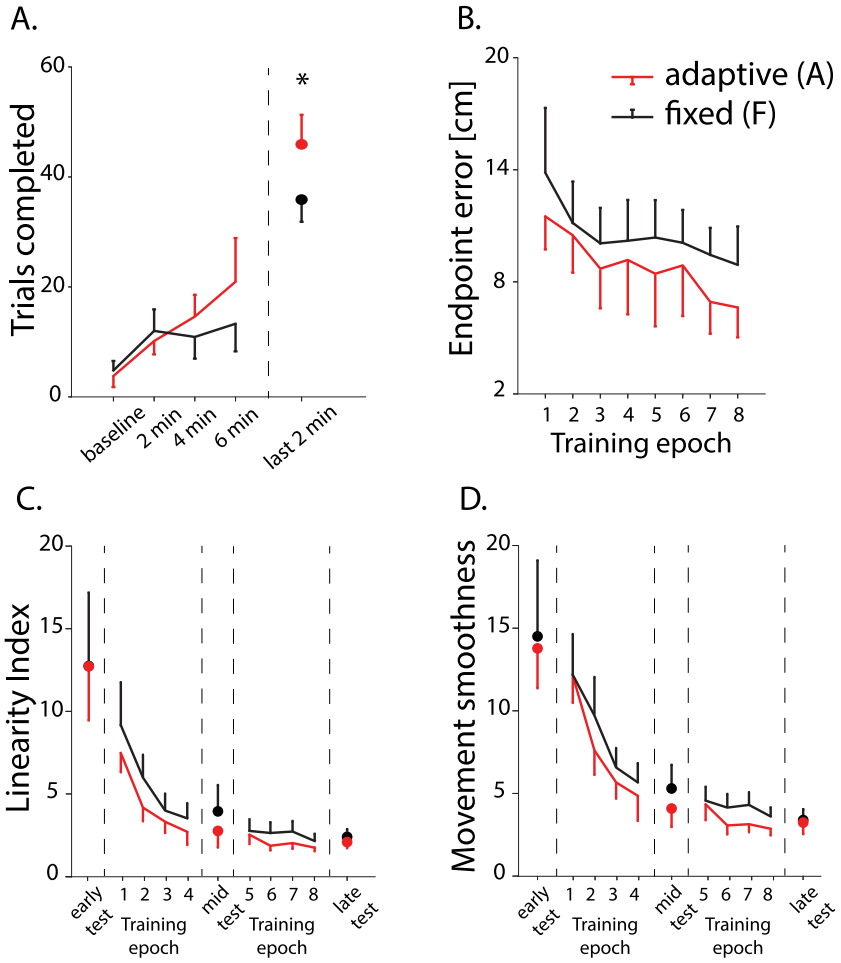



**Fig. 5.** Performance metrics for adaptive (red) and fixed (black) group. Panel A: Number of trials completed during baseline, after two, four, six minutes following baseline and during the final two minutes of training. The asterisk represents a significant difference between groups during the last 2 min of training. Panel B: Endpoint error during blind trials of each training epoch. Panel C-D: Linearity index and movement smoothness values during each training and test epoch. Mean values across participants are plotted with 95% confidence interval.

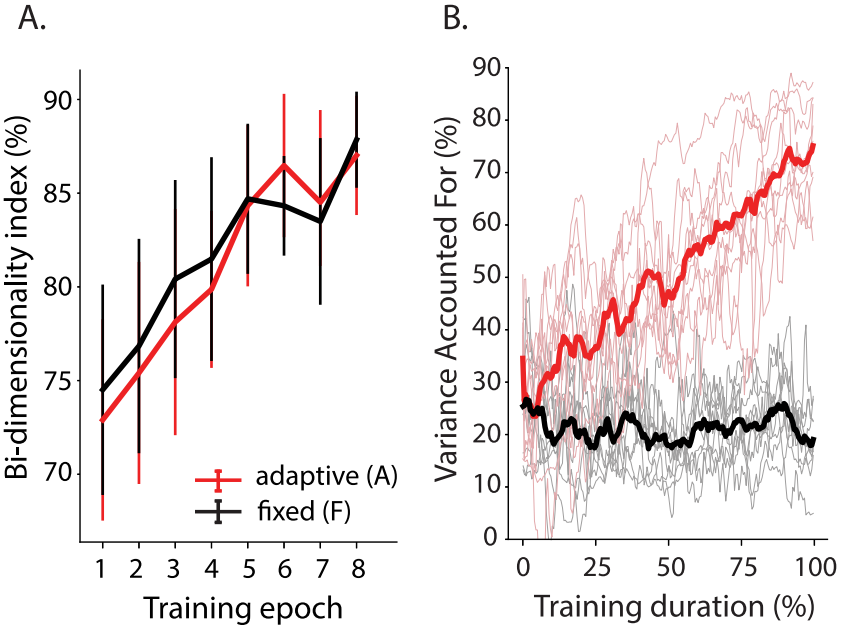



**Fig. 6.** Panel A: Bi-dimensionality index values for adaptive (red) and static (black) group during each training epoch. Mean values across participants are plotted with 95% confidence interval. Panel B: Variance accounted for (VAF) values for adaptive (red) and static (black) group during each batch of the training duration. Mean across subjects is shown as a bold line for both groups.

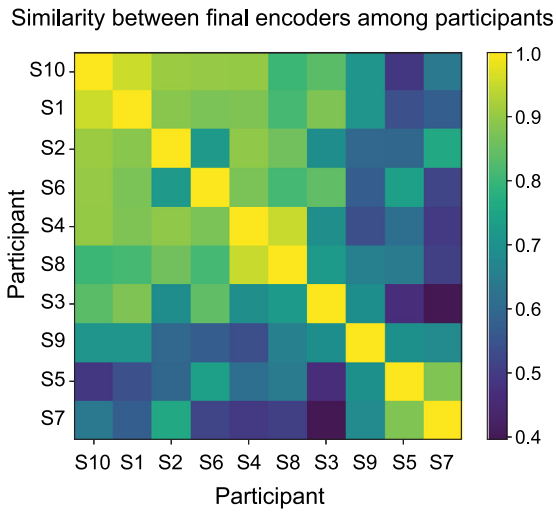



**Fig. 7.** Similarity matrix representing the similarity between final encoders among participants. The higher the value of a cell, the more similar the final encoders of the two participants of that cell. The matrix has been ranked to cluster participants with similar final encoders.

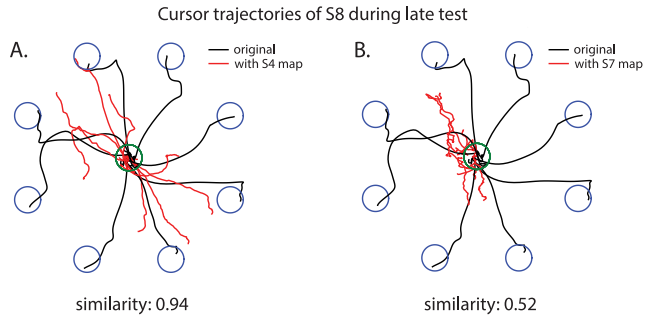



**Fig. 8.** Cursor trajectories of one participant (S8, black lines) during the final test epoch. We took the IMU data of S8 recorded during the final test epoch and applied the final encoder of a participant with whom S8 had a high similarity (S4, red lines, Panel A) and a low similarity (S7, red lines, Panel B).

substitute the final AE map that a participant used in the late test phase with that of another participant whose map had a comparable structure, the same participant would have been able to cover the target space much more consistently (Fig. 8A) than with a map whose similarity was not as pronounced (Fig. 8B).

## 5. Discussion

In this study we proposed an adaptive platform based on the use of an iterative non-linear autoencoder to implement unsupervised tracking of user's manifold for improving the ease-of-use of a human–machine interface. First and foremost, our results support the use of a non-linear AE as a proficient control map within the body–machine interface scheme. Moreover, the adaptive approach led to an increased representational efficiency of the interface decoder while concurrently increasing users' task-related performance, both in terms of number of trials completed over time and accuracy during reaching of blind trials. This result suggests that the online co-adaptation process encourages the development of a more accurate internal model. Importantly, the proposed approach has three salient features that makes it appealing in many applications other than the one tested here: (i) it cancels the cost of interrupting the operation of the device to perform decoder recalibration, (ii) as no information about the state of the task and/or intended task goals is needed, the manifold tracking algorithm can be applied to a great variety of contexts, iii) it does not rely on the existence of a stable neural or movement manifold to compensate for decoder instabilities, allowing it to be applied in the earliest stages of interface operation, when the formation of new neural strategies is still on-going.

### 5.1. Autoencoder networks can proficiently control low-dimensional devices

There is an increasing enthusiasm about using autoencoder networks in the field of human–machine interfaces (HMIs). AEs give the freedom of choosing the type of network architecture and level of complexity (*e.g.*, linear/non-linear, generative/convolutional/recurrent, number of hidden layers and neurons per layer), thus potentially allowing to better match the degree of complexity of the input signal and hence allow for a great variety of applications. For instance, AE-based approaches have recently been employed for adversarial and variational domain adaptation (Farshchian et al., 2018; Hsu et al., 2017) and for extracting precise estimates of neural dynamics (Pandarinath et al., 2018). However, applicability of non-linear AEs for the control of low dimensional devices have seen limited efforts.

The use of AEs, or more specifically of their encoder sub-network, as a forward map in a BoMI has been first proposed by Pierella et al. (2018). In that study, an AE encoder was used to map muscle activities recorded via EMG to the coordinates of a computer cursor, similarly to the case considered here. However, there is a crucial difference between these two implementations. In Pierella et al. (2018), the AE cost function was modified by adding a constraint to Eq. (2) to force the latent space dimensions to follow a hierarchy in terms of their variance, mimicking PCA (hierarchical autoencoder Scholz et al., 2008). The inclusion of this constraint may limit the reconstruction power of the autoencoder. Hence, the approach is not optimal for accurately estimating the latent space defined by BoMI user movements, with potential repercussions on the ease-of-use of the interface. In order to avoid this potential pitfall and maximize the variance accounted for by the latent representation (Portnova-Fahreeva et al., 2020), here we decided to use a "vanilla" autoencoder that performs DR by minimizing Eq. (2) without any added constraint. The AE architecture used in our study was very parsimonious, since the addition of more parameters did not increase the variance explained on the movement calibration data set. Even with our parsimonious choice, the AE in Fig. 1 allowed retaining 93% of the variance of the calibration data set in its latent encoding, while a linear DR algorithm (PCA) would have accounted only for 72% of the original variance on the same data set.

Few other studies have explored the representational power of AEs in HMIs. In the work of Vujaklija et al. (2018), an AE was used to map the EMG activity of forearm muscles during instructed wrist movements with one degree of freedom (DoF) (*i.e.*, flexion/extension, or radial/ulnar deviation) into cursor displacement along a line. Losey et al. (2020) used an AE to extract non-linear motion primitives from guided movements of a robotic arm. The users controlled the robot by selecting latent actions with a joystick and the AE decoder sub-network mapped the selected action back into the high DoF robot movement.

However, while the use of non-linear AEs is ideal when modelling any process that has a non-linear structure as in the previous examples, their use in the closed-loop schema of an HMI might result challenging. On the one hand, having a better estimate of the BoMI user's intentions might allow the user to generate more accurate motor commands. On the other, non-linearity comes with the loss of a major advantage of linear control — the possibility to obtain a full repertoire of actions from the direct summation of simpler actions. Nevertheless, this

study found that learning to operate a non-linear interface is not only possible, but comparable to what has been observed with a linear interface (De Santis & Mussa-Ivaldi, 2020), thus providing a proof of concept for the use of non-linear autoencoders for human–machine interaction.

In this study, we also took a step forward and utilized non-linear AEs in an iterative way for closed-loop adaptation with a learning agent. The online update of AEs posed different challenges, such as the hyperparameter tuning that we discuss in Section 5.2. Nevertheless, we showed that iteratively updating an autoencoder used as control map to track user's latent manifold is not only successful (Fig. 6B) but led to an increase in user's task-related performance when operating the interface.

### 5.2. Map adaptation converges during offline test

When dealing with any learning algorithm, the common problem of choosing a set of optimal parameters is typically referred to as (hyper)parameter tuning (Claesen & De Moor, 2015). In our case, there were three main parameters that required such tuning: the learning rate $\alpha$, the number of steps $\beta$ of BGD and the size of the history-batch. Each parameter influenced the a-AE tracking capability of user's movement manifold.

First and foremost, we had to define which time frame of user's previous movements could be considered representative of its movement distribution. In this sense, the advantage of using a short-term memory is that the update operation would be the least computationally expensive. However, the a-AE memory needs to contain a sufficiently comprehensive description of user's movement statistics to achieve the desired convergence rate target. Hence, a short-term memory is the most prone to being locally biased especially in the initial phase of practice with the interface, as the movement variability may be confined to a small region of space. This may prevent the adaptive algorithm from converging within an acceptable time frame (see 5*$tau$ in Fig. 4A). Thus, one would want to consider a memory whose size is increased (*i.e.*, a mid-term or long-term memory). Increasing the memory batch expands the time horizon of user movements accounted for by the a-AE, making the online training more representative of user's evolving distribution and less prone to overfit (or, equivalently, smoother), at the expense of a computationally more expensive update. The choice of one over the other was motivated by looking at the concurrent value of learning rate.

Results showed that a learning rate of $\alpha = 10^{-5}$ prevented the algorithm from converging within an acceptable time frame. Therefore, the a-AE would not have been able to keep track of the evolving distribution of user's movements. With $\alpha = 10^{-3}$, the adaptive algorithm fit user's movements in the fastest time frame (25s with a long-term history batch of 60s, Fig. 4A). Ideally, the adaptive algorithm should wait for the user to develop a meaningful motor strategy before converging. However, during the first stages of learning, BoMI users typically complete only one reaching trial within the time frame of 20/30s. As mentioned by Müller et al. (2017), when the machine learns too fast, the co-adaptive process is unstable and unable to converge. The additional metrics that we computed seem to support this claim, as the rate of changes (Fig. 4B) and the static jump (Fig. 4C) increased proportionally with the network learning rate. On the other hand, with $\alpha = 10^{-4}$ (and a long-term memory of 60s), the algorithm managed to reach the set point of convergence within 160 s, which was considerably closer to the target value of 2 min.

As a result, we opted for the intermediate learning rate ($\alpha = 10^{-4}$) and a long-term history batch ($k = 60$ s). This choice was further motivated by the work of Dangi et al. (2013) and Orsborn et al. (2011, 2012), recommending time scale of updates of 1 min for applications to adaptive brain–machine interface, and those of Danziger et al. (2009), Davidson and Wolpert (2003), Golub et al. (2018) and Orban De Xivry and Lefèvre (2015) showing that interfaces that changes too abruptly during online operation are detrimental for users' learning.

We ran the sensitivity tuning with a prefixed (and arbitrary) value of $\beta = 10$. Since this value allowed to satisfy our tuning goals, we decided to exclude this parameter from the tuning procedure and chose it as our final parameter for the online reaching test.

### 5.3. Map adaptation leads to superior performance

With the proposed online adaptive algorithm, participants significantly outperformed the fixed group in terms of number of trials completed over time. Indeed, by the end of the training, the adaptive group reached a significantly higher number of targets than the fixed group (Fig. 5A). It is interesting to notice that, while both groups unsurprisingly started with the same level of performance, the contribution of the a-AE started improving the performance of the adaptive group already after four minutes. The performance difference between the two groups became increasingly significant over time, finally reaching its peak in the last two minutes of training. These results suggest how, after an initial period of co-adjustments between the BoMI users and the interface, the adaptation of the latter towards user's movement manifold successfully helped participant's ease-of-use of the interface. Moreover, we found how the a-AE promoted the development of an internal representation of the cursor space, as participants were able to identify the position of targets more precisely during the blind trials (Fig. 5B, with a nearly approached significance threshold). This result is consistent with that obtained by our previous study of an online co-adaptation with a linear interface (De Santis & Mussa-Ivaldi, 2020). Interestingly, both the fixed and the adaptive group organized their movements towards a structure that gradually resembled that of a two-dimensional manifold as they learned how to operate the interface (Fig. 6A). This confirms our hypothesis that participants, through motor learning, distributed their motions so as to match the dimensionality of the sensory feedback, in line with previous studies (Mosier et al., 2005; Ranganathan et al., 2014).

Remarkably, the proposed adaptive algorithm was successful in tracking and tailoring its user's movements (Fig. 6B). This further validates the choice of the a-AE parameters made after the sensitivity tuning and, together with the reported improvement in task-related performance, confirms our initial hypothesis that human–machine interaction can be promoted if the interface low dimensional output and the human's manifold are isomorphic. The use of an adaptive interface that tracked participants' manifold over time allowed shaping their movement distribution in a manifold resembling that of the interface itself. As a result, we speculate that the Adaptive group converged to a more efficient inverse model of the AE encoder. The finding that the Adaptive group was less dependent on visual feedback with respect to the Fixed group supports this idea. We believe that the Fixed group was still able to efficiently operate the interface even in the presence of more inaccurate estimates of the inverse AE encoder because of the redundant nature of our interface. Namely, participants could have used different motor strategies to complete the task. Therefore, learning of an accurate predictive component was not necessary as long as participants could have relied on the visual feedback to compensate for aiming inaccuracies. Moreover, the Static group could have also attained high accuracy from the prolonged exposure to the same map.

### 5.4. Map adaptation is guided by the individual's learning trajectory

If on the one hand the a-AE was successful in tracking the movement manifold of each participant, on the other we noted that the changes in the encoder were not consistent across participants. Namely, we could divide the BoMI users into two main groups: those who converged towards a similar encoder structure at the end of the training (S1, S2, S4, S6, S8, S10 - Fig. 7) and those whose maps did not converge to a particular similarity value (S3, S5, S7, S9 - Fig. 7). All the participants were able to reach a satisfactory level of performance regardless of whether the map was converging towards a particular structure. Furthermore, as mentioned before, in addition to being adaptive, the redundant character of the interface allowed each participant to complete the task with a different strategy. We speculate that the final encoder was merely a result of the learning trajectory of each participant. Indeed, if two participants were to exhibit the same movement dynamics (*i.e.*, strategy) from the start to the end of the practice, it is reasonable to assume that they would have converged towards the same encoder. Since this was not always the case, we concluded that some participants effectively learned the task in a different way. It is important to remark that the a-AE was able to guide them to an efficient resolution of the task regardless of the inverse model employed by the participant. This is a remarkable characteristic that increases the generalization capability of the proposed interface.

### 5.5. Perspective on current adaptive interfaces

The problem of building adaptive interfaces is raising increasing interest in the realm of human–machine interactions. Here we focused on the development of an adaptive interface that provides a seamless interaction with the user during online operation of the interface. In the field of BMIs, the closed-loop adaptation of the interface is driven by some policies associated with the user's known movement intention (Vidaurre et al., 2010). This results in supervised adaptation as, for example, it would require the user to perform pre-selected movements to guide the update of the interface parameters (Dangi et al., 2014; Orsborn et al., 2012). Another possible approach to adapt the interface so as to account for user's motor strategies could stem from Reinforcement Learning (RL) (Sutton & Barto, 2018), where a software agent continually interacts with an environment and take actions in order to maximize some reward. Previous studies (DiGiovanna et al., 2009; Mahmoudi et al., 2008; Sanchez et al., 2009) have used RL-inspired algorithms to modify the agent's (the interface) behaviour according to what was considered desirable for the user. The RL approach, however, still requires the definition of a value function in order to assign a reward to an observed action.

To improve the generalizability of the adaptive interface across tasks, an unsupervised approach is most appropriate. If on the one hand Mehring et al. proved that an unsupervised co-adaptation is theoretically possible (Gürel & Mehring, 2012), on the other there is still a clear gap of knowledge in the implementation of an unsupervised adaptation concurrent with the operation of the interface. In this sense, a recent work (Degenhart et al., 2020) proved that an unsupervised re-alignment of intrinsic manifold of neural activities could stabilize interface performance in the presence of recording instabilities. There are similarities between the approach described in our study and theirs, as both share the basic rationale of exploiting latent manifolds for updating the interface. However, unlike in Degenhart et al. (2020), here we are concerned with tracking a time-varying manifold, rather than stabilizing it against artefactual changes. Therefore, our co-adaptation approach is intended to operate immediately, without waiting for the users to having developed a stable motor strategy.

We want to stress that the this study represents a novel approach for designing an adaptive interface, as it avoids constraints that other state-of-the-art interfaces commonly have, such as the need of interrupting the operation of the device, of relying on information about the state of the task, or relying on the existence of a stable neural or movement manifold. As such, directly comparing the performance of our interface against those of other interfaces with completely different nature is not straightforward. Perhaps a comparison can be made with an earlier study performed by our own group (De Santis & Mussa-Ivaldi, 2020), in which a linear model, iterative PCA, was used to align the space of body movements with the space encoded by the interface. However, the experimental setups differed in the placement of the inertial sensors (two IMUs on the arms in this study, four IMUs – two on the arms and two on the forearms – in the previous study). Nonetheless, the study presented in this manuscript confirmed some of the findings of our previous one, in particular the ability of the adaptive interface to lead to the development of a more faithful internal representation of the control problem (Fig. 6B). Furthermore, differently from the linear case, we found that, with the proposed non-linear adaptive interface, participants were able to significantly outperform those practising with a static map in terms of target acquisition rate. We believe that this difference might be due to the superiority of non-linear models in estimating non-linear latent manifolds (Tenenbaum et al., 2000) such as those spanned by upper body kinematics. This has further implications for enhancing the generalizability of the interface proposed in this study, as it might be more appropriate when dealing with manifolds derived from processes such as neural recordings, as well as when developing a controller for multi-degrees-of-freedom devices.

### 5.6. Limitations

The choice of the learning rate $\alpha$ of the network was motivated by the tuning procedure we ran as a first stage of this study. However, our implementation of the online co-adaptation did not consider the rate of the BoMI user learning the task. In this sense, an online co-adaptation between the learning rate $\alpha$ of the network and the learning rate of the user operating the BoMI might improve the ease-of-use of the interface itself. This could be achieved by implementing a policy that adapts $\alpha$ accordingly to the user's learning time scale. A higher value of $\alpha$ might be beneficial during the first part of the learning curve (cognitive stage Fitts & Posner, 1967). During this stage of learning, humans typically present high-movement, and consequently, high-performance variability. As stated before, however, there is a dangerous threshold, after which forcing people to explore more will, most likely, prevent them from learning the task at all. We did not find any study that deals with the problem of developing an algorithm that accounts for this threshold, and, as a result, further efforts in this sense are required. Going back to the policy design, as soon as the policy registers an increase in user's performance, the value of $\alpha$ could be decreased to encourage a strategy consolidation. The policy might eventually set the learning rate of the network to a minimum value when user's performance is about to plateau. Our study, however, suggested that letting the interface continuously adapting, even after the user reached a sufficient capability of controlling the cursor, did not disrupt motor learning. In other words, the proposed interface found its own stability without any supervision. We assume that this was a result of the design choice to consider an extensive portion of user's latest movements. Thus, as long as the statistical distribution of those movements was not changing, the encoder was not changing substantially, de facto autoregulating itself.

## Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

## Funding

This work was supported by the Marie Curie Integration [Grant FP7- PEOPLE-2012-CIG-334201], the Ministry of Science and Technology, Israel (Joint Israel–Italy lab in Biorobotics Artificial somatosensorial for humans and humanoids), the National Science Foundation [Grant 1632259], the NIDILRR [Grant 90REGE0005-01], the NICHHD [Grant 5R01HD072080], NIBIB, USA (Grant No. R01 EB024058-03) and the European Union Horizon 2020 research and innovation program under the Marie Sklodowska-Curie, project REBoT, [G.A. No 750464].

## References

Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M., Kudlur, M., Levenberg, J., Monga, R., Moore, S., Murray, D. G., Steiner, B., Tucker, P., Vasudevan, V., Warden, P., .... Zheng, X. (2016). TensorFlow: A system for large-scale machine learning. In *12th {USENIX} symposium on operating systems design and implementation* (pp. 265–283). https://www.usenix.org/conference/osdi16/technical-sessions/presentation/abadi.

Abdollahi, F., Farshchiansadegh, A., Pierella, C., Seáñez González, I., Thorp, E., Lee, M. H., Ranganathan, R., Pedersen, J., Chen, D., Roth, E., Casadio, M., & Mussa-Ivaldi, F. (2017). Body-machine interface enables people with cervical spinal cord injury to control devices with available body movements: Proof of concept. *Neurorehabilitation and Neural Repair*, *31*(5), 487–493. http://dx.doi.org/10.1177/1545968317693111.

Barrese, J. C., Rao, N., Paroo, K., Triebwasser, C., Vargas-Irwin, C., Franquemont, L., & Donoghue, J. P. (2013). Failure mode analysis of silicon-based intracortical microelectrode arrays in non-human primates. *Journal of Neural Engineering*, *10*(6), http://dx.doi.org/10.1088/1741-2560/10/6/066014.

Bengio, Y. (2012). Deep learning of representations for unsupervised and transfer learning. In *Proceedings of ICML workshop on unsupervised and transfer learning (vol. 7)* (pp. 17–36).

Carmena, J. M., Lebedev, M. A., Crist, R. E., O'Doherty, J. E., Santucci, D. M., Dimitrov, D. F., Patil, P. G., Henriquez, C. S., & Nicolelis, M. A. L. (2003). Learning to control a brain-machine interface for reaching and grasping by primates. *PLoS Biology*, *1*(2), 193–208. http://dx.doi.org/10.1371/journal.pbio.0000042.

Casadio, M., Pressman, A., Fishbach, A., Danziger, Z., Acosta, S., Chen, D., Tseng, H. Y., & Mussa-Ivaldi, F. A. (2010). Functional reorganization of upper-body movement after spinal cord injury. *Experimental Brain Research*, *207*(3–4), 233–247. http://dx.doi.org/10.1007/s00221-010-2427-8.

Casadio, M., Ranganathan, R., & Mussa-Ivaldi, F. A. (2012). The body-machine interface: A new perspective on an old theme. *Journal of Motor Behavior*, *44*(6), 419–433. http://dx.doi.org/10.1080/00222895.2012.700968.

Claesen, M., & De Moor, B. (2015). Hyperparameter search in machine learning. ArXiv Preprint arXiv:1502.02127.

Dangi, S., Gowda, S., Moorman, H. G., Orsborn, A. L., So, K., Shanechi, M., & Carmena, J. M. (2014). Continuous closed-loop decoder adaptation with a recursive maximum likelihood algorithm allows for rapid performance acquisition in brain-machine interfaces. *Neural Computation*, *26*(9), 1811–1839. http://dx.doi.org/10.1162/NECO.

Dangi, S., Orsborn, A. L., Moorman, H. G., & Carmena, J. M. (2013). Design and analysis of closed-loop decoder adaptation algorithms for brain-machine interfaces. *Neural Computation*, *25*(7), 1693–1731.

Danziger, Z., Fishbach, A., & Mussa-Ivaldi, F. A. (2009). Learning algorithms for human–machine interfaces. *IEEE Transactions on Biomedical Engineering*, *56*(5), 1502–1511.

Davidson, P. R., & Wolpert, D. M. (2003). Motor learning and prediction in a variable environment. *Current Opinion in Neurobiology*, *13*(2), 232–237. http://dx.doi.org/10.1016/S0959-4388(03)00038-2.

Dayan, E., & Cohen, L. G. (2011). Neuroplasticity subserving motor skill learning. *Neuron*, *72*(3), 443–454. http://dx.doi.org/10.1016/j.neuron.2011.10.008.

De Santis, D., Dzialecka, P., & Mussa-Ivaldi, F. A. (2018). Unsupervised coadaptation of an assistive interface to facilitate sensorimotor learning of redundant control. In *Proceedings of the IEEE RAS and EMBS international conference on biomedical robotics and biomechatronics* (pp. 801–806). http://dx.doi.org/10.1109/BIOROB.2018.8487912.

De Santis, D., & Mussa-Ivaldi, F. A. (2020). Guiding functional reorganization of motor redundancy using a body-machine interface. *Journal of NeuroEngineering and Rehabilitation*, 1–17.

Degenhart, A. D., Bishop, W. E., Oby, E. R., Tyler-Kabara, E. C., Chase, S. M., Batista, A. P., Yu, B. M., & Byron, M. Y. (2020). Stabilization of a brain–computer interface via the alignment of low-dimensional spaces of neural activity. *Nature Biomedical Engineering*, 1–14. http://dx.doi.org/10.1038/s41551-020-0542-9.

DiGiovanna, J., Mahmoudi, B., Fortes, J., Principe, J. C., & Sanchez, J. C. (2009). Coadaptive brain-machine interface via reinforcement learning. *IEEE Transactions on Biomedical Engineering*, *56*(1), 54–64. http://dx.doi.org/10.1109/TBME.2008.926699.

Donati, A. R. C., Shokur, S., Morya, E., Campos, D. S. F., Moioli, R. C., Gitti, C. M., Augusto, P. B., Tripodi, S., Pires, C. G., Pereira, G. A., Brasil, F. L., Gallo, S., Lin, A. A., Takigami, A. K., Aratanha, M. A., Joshi, S., Bleuler, H., Cheng, G., Rudolph, A., & Nicolelis, M. A. L. (2016). Long-term training with a brain-machine interface-based gait protocol induces partial neurological recovery in paraplegic patients. *Scientific Reports*, *6*(July), 1–16. http://dx.doi.org/10.1038/srep30383.

Downey, J. E., Schwed, N., Chase, S. M., Schwartz, A. B., & Collinger, J. L. (2018). Intracortical recording stability in human brain-computer interface users. *Journal of Neural Engineering*, *15*(4), http://dx.doi.org/10.1088/1741-2552/aab7a0.

Farshchian, A., Gallego, J. A., Cohen, J. P., Bengio, Y., Miller, L. E., & Solla, S. A. (2018). Adversarial domain adaptation for stable brain-machine interfaces. (pp. 1–14). ArXiv Preprint arXiv:1810.00045.

Farshchiansadegh, A., Abdollahi, F., Chen, D., Lee, M. H., Pedersen, J., Pierella, C., Roth, E. J., Gonzalez, I. S., Thorp, E. B., & Mussa-Ivaldi, F. A. (2014). A body machine interface based on inertial sensors. In *2014 36th annual international conference of the IEEE engineering in medicine and biology society* (pp. 6120–6124). http://dx.doi.org/10.1109/EMBC.2014.6945026.

Fitts, P. M., & Posner, M. I. (1967). Human performance. In *Human performance*. Brooks/Cole.

Gallego, J. A., Perich, M. G., Chowdhury, R. H., Solla, S. A., & Miller, L. E. (2020). Long-term stability of cortical population dynamics underlying consistent behavior. *Nature Neuroscience*, *23*(FebruarY), http://dx.doi.org/10.1038/s41593-019-0555-4.

Ganguly, K., & Carmena, J. M. (2009). Emergence of a stable cortical map for neuroprosthetic control. *PLoS Biology*, *7*(7), Article e1000153. http://dx.doi.org/10.1371/journal.pbio.1000153.

Golub, M. D., Sadtler, P. T., Oby, E. R., Quick, K. M., Ryu, S. I., Tyler-Kabara, E. C., Batista, A. P., Chase, S. M., & Yu, B. M. (2018). Learning by neural reassociation. *Nature Neuroscience*, *21*(4), 607–616. http://dx.doi.org/10.1038/s41593-018-0095-3.

Seáñez González, I., Pierella, C., Farshchiansadegh, A., Thorp, E. B., Wang, X., Parrish, T., & Mussa-Ivaldi, F. A. (2016). Body-machine interfaces after spinal cord injury: Rehabilitation and brain plasticity. *Brain Sciences*, *6*(4), 1–19. http://dx.doi.org/10.3390/brainsci6040061.

Gürel, T., & Mehring, C. (2012). Unsupervised adaptation of brain-machine interface decoders. *Frontiers in Neuroscience*, *6*(NOV), 1–18. http://dx.doi.org/10.3389/fnins.2012.00164.

Hsu, W.-N., Zhang, Y., & Glass, J. (2017). Unsupervised domain adaptation for robust speech recognition via variational autoencoder-based data augmentation. In *2017 IEEE automatic speech recognition and understanding workshop* (pp. 16–23).

Kandel, E. R., Schwartz, J. H., Jessell, T. M., Siegelbaum, S., Hudspeth, A. J., & Mack, S. (2000). In Eric R. Kandel, James H. Schwartz, & Thomas M. Jessell (Eds.), *Principles of neural science (vol. 4)*. McGraw-hill New York.

Kingma, D. P., & Ba, J. L. (2015). Adam: A method for stochastic optimization. In *3rd international conference on learning representations, conference track proceedings* (pp. 1–15).

Kornblith, S., Norouzi, M., Lee, H., & Hinton, G. (2019). Similarity of neural network representations revisited. ArXiv Preprint arXiv:1905.00414.

Kramer, M. A. (1991). Nonlinear principal component analysis using autoassociative neural networks. *AIChE Journal*, *37*(2), 233–243. http://dx.doi.org/10.1002/aic.690370209.

Losey, D. P., Srinivasan, K., Mandlekar, A., Garg, A., & Sadigh, D. (2020). Controlling assistive robots with learned latent actions. In *2020 IEEE international conference on robotics and automation* (pp. 378–384). http://arxiv.org/abs/1909.09674.

Mahmoudi, B., DiGiovanna, J., Principe, J. C., & Sanchez, J. C. (2008). Co-adaptive learning in brain-machine interfaces. In *Brain Inspired Cognitive Systems* (pp. 1–5). Sao Luis, Brazil.

Mawase, F., Uehara, S., Bastian, A. J., & Celnik, P. (2017). Motor learning enhances use-dependent plasticity. *The Journal of Neuroscience*, *37*(10), 2673–2685. http://dx.doi.org/10.1523/JNEUROSCI.3303-16.2017.

Miehlbradt, J., Cherpillod, A., Mintchev, S., Coscia, M., Artoni, F., Floreano, D., & Micera, S. (2018). Data-driven body-machine interface for the accurate control of drones. *Proceedings of the National Academy of Sciences of the United States of America*, *115*(31), 7913–7918. http://dx.doi.org/10.1073/pnas.1718648115.

Morcos, A. S., Raghu, M., & Bengio, S. (2018). Insights on representational similarity in neural networks with canonical correlation. In *Advances in neural information processing systems* (pp. 5727–5736).

Mosier, K. M., Scheidt, R. A., Acosta, S., & Mussa-Ivaldi, F. A. (2005). Remapping hand movements in a novel geometrical environment. *Journal of Neurophysiology*, *94*(6), 4362–4372. http://dx.doi.org/10.1152/jn.00380.2005.

Müller, J. S., Vidaurre, C., Schreuder, M., Meinecke, F. C., Von Bünau, P., & Müller, K.-R. (2017). A mathematical model for the two-learners problem. *Journal of Neural Engineering*, *14*(3), 36005. http://dx.doi.org/10.1088/1741-2552/aa620b.

Oby, E. R., Golub, M. D., Hennig, J. A., Degenhart, A. D., Tyler-Kabara, E. C., Yu, B. M., Chase, S. M., & Batista, A. P. (2019). New neural activity patterns emerge with long-term learning. *Proceedings of the National Academy of Sciences of the United States of America*, *116*(30), 15210–15215. http://dx.doi.org/10.1073/pnas.1820296116.

Orban De Xivry, J. J., & Lefèvre, P. (2015). Formation of model-free motor memories during motor adaptation depends on perturbation schedule. *Journal of Neurophysiology*, *113*(7), 2733–2741. http://dx.doi.org/10.1152/jn.00673.2014.

Orsborn, A. L., Dangi, S., Moorman, H. G., & Carmena, J. M. (2011). Exploring time-scales of closed-loop decoder adaptation in brain-machine interfaces. In *2011 annual international conference of the IEEE engineering in medicine and biology society* (pp. 5436–5439).

Orsborn, A. L., Dangi, S., Moorman, H. G., Carmena, J. M., & Member, S. (2012). Closed-loop decoder adaptation on intermediate time-scales facilitates rapid BMI performance improvements independent of decoder initialization conditions. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, *20*(4), 468–477.

Orsborn, A. L., Moorman, H. G., Overduin, S. A., Shanechi, M. M., Dimitrov, D. F., & Carmena, J. M. (2014). Closed-loop decoder adaptation shapes neural plasticity for skillful neuroprosthetic control. *Neuron*, *82*(6), 1380–1393. http://dx.doi.org/10.1016/j.neuron.2014.04.048.

Oweiss, K. G., & Badreldin, I. S. (2015). Neuroplasticity subserving the operation of brain-machine interfaces. *Neurobiology of Disease*, *83*, 161–171. http://dx.doi.org/10.1016/j.nbd.2015.05.001.

Pandarinath, C., O'Shea, D. J., Collins, J., Jozefowicz, R., Stavisky, S. D., Kao, J. C., Trautmann, E. M., Kaufman, M. T., Ryu, S. I., Hochberg, L. R., Henderson, J. M., Shenoy, K. V., Abbott, L. F., & Sussillo, D. (2018). Inferring single-trial neural population dynamics using sequential auto-encoders. *Nature Methods*, *15*(10), 805–815. http://dx.doi.org/10.1038/s41592-018-0109-9.

Pierella, C., Abdollahi, F., Thorp, E., Farshchiansadegh, A., Pedersen, J., Seáñez González, I., Mussa-Ivaldi, F. A., & Casadio, M. (2017). Learning new movements after paralysis: Results from a home-based study. *Scientific Reports*, *7*(1), 1–11. http://dx.doi.org/10.1038/s41598-017-04930-z.

Pierella, C., Sciacchitano, A., Farshchiansadegh, A., Casadio, M., & Mussa-Ivaldi, S. A. (2018). Linear vs non-linear mapping in a body machine interface based on electromyographic signals. In *Proceedings of the IEEE RAS and EMBS international conference on biomedical robotics and biomechatronics* (pp. 162–166). http://dx.doi.org/10.1109/BIOROB.2018.8487185.

Portnova-Fahreeva, A. A., Rizzoglio, F., Nisky, I., & Casadio, M. (2020). Linear and non-linear techniques on full hand kinematics. *Frontiers in Bioengineering and Biotechnology*, *8*(May), 1–18. http://dx.doi.org/10.3389/fbioe.2020.00429.

Raghu, M., Gilmer, J., Yosinski, J., & Sohl-Dickstein, J. (2017). SVCCA: Singular vector canonical correlation analysis for deep learning dynamics and interpretability. In *Advances in neural information processing systems* (pp. 6077–6086).

Ranganathan, R., Wieser, J., Mosier, K. M., Mussa-Ivaldi, F. A., & Scheidt, R. A. (2014). Learning redundant motor tasks with and without overlapping dimensions: facilitation and interference effects. *Journal of Neuroscience*, *34*(24), 8289–8299.

Rizzoglio, F., Pierella, C., Santis, D., De, Mussa-Ivaldi, F. A., De Santis, D., Mussa-Ivaldi, F. A., & Casadio, M. (2020). A hybrid body-machine interface integrating signals from muscles and motions. *Journal of Neural Engineering*, http://dx.doi.org/10.1088/1741-2552/ab9b6c.

Ruder, S. (2016). An overview of gradient descent optimization algorithms. (pp. 1–14). http://arxiv.org/abs/1609.04747.

Sanchez, J. C., Mahmoudi, B., DiGiovanna, J., & Principe, J. C. (2009). Exploiting co-adaptation for the design of symbiotic neuroprosthetic assistants. *Neural Networks*, *22*(3), 305–315. http://dx.doi.org/10.1016/j.neunet.2009.03.015.

Scholz, M., Fraunholz, M., & Selbig, J. (2008). Nonlinear principal component analysis: Neural network models and applications. *Lecture Notes in Computational Science and Engineering*, *58*, 44–67. http://dx.doi.org/10.1007/978-3-540-73750-6_2.

Shadmehr, R., & Mussa-Ivaldi, F. A. (1994). Adaptive representation of dynamics during learning of a motor task. *Journal of Neuroscience*, *14*(5), 3208–3224.

Shadmehr, R., Smith, M. A., & Krakauer, J. W. (2010). Error correction, sensory prediction, and adaptation in motor control. *Annual Review of Neuroscience*, *33*(1), 89–108. http://dx.doi.org/10.1146/annurev-neuro-060909-153135.

Shanechi, M. M. (2017). Brain-machine interface control algorithms. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, *25*(10), 1725–1734. http://dx.doi.org/10.1109/TNSRE.2016.2639501.

Shenoy, K. V., & Carmena, J. M. (2014). Combining decoder design and neural adaptation in brain-machine interfaces. *Neuron*, *84*(4), 665–680. http://dx.doi.org/10.1016/j.neuron.2014.08.038.

Sutton, R. S., & Barto, A. G. (2018). *Reinforcement learning: An introduction*. MIT Press.

Tenenbaum, J. B., De Silva, V., & Langford, J. C. (2000). A global geometric framework for nonlinear dimensionality reduction. *Science*, *290*(5500), 2319–2323. http://dx.doi.org/10.1126/science.290.5500.2319.

Thompson, B. (2005). Canonical correlation analysis. In *Encyclopedia of statistics in behavioral science*. American Cancer Society, http://dx.doi.org/10.1002/0470013192.bsa068.

Vidaurre, C., Sannelli, C., Müller, K.-R., & Blankertz, B. (2010). Machine-learning-based coadaptive calibration for brain-computer interfaces. *Neural Computation*, *23*(3), 791–816. http://dx.doi.org/10.1162/NECO_a_00089.

Vujaklija, I., Shalchyan, V., Kamavuako, E. N., Jiang, N., Marateb, H. R., & Farina, D. (2018). Online mapping of EMG signals into kinematics by autoencoding. *Journal of NeuroEngineering and Rehabilitation*, *15*(1), http://dx.doi.org/10.1186/s12984-018-0363-1.

Wei, Y., Bajaj, P., Scheldt, R., & Patton, J. (2005). Visual error augmentation for enhancing motor learning and rehabilitative relearning. *2005*, In *Proceedings of the 2005 IEEE 9th international conference on rehabilitation robotics* (pp. 505–510). http://dx.doi.org/10.1109/ICORR.2005.1501152.

Wold, S., Esbensen, K., & Geladi, P. (1987). Principal component analysis. *Chemometrics and Intelligent Laboratory Systems*, http://dx.doi.org/10.1016/0169-7439(87)80084-9.