

Stabilizing brain-computer interfaces through alignment of latent dynamics

Received: 19 February 2025

Accepted: 24 April 2025

Published online: 19 May 2025

 Check for updates

Brianna M. Karpowicz¹, Yahia H. Ali¹, Lahiru N. Wimalasena¹, Andrew R. Sedler^{1,2}, Mohammad Reza Keshtkaran¹, Kevin Bodkin ³, Xuan Ma ³, Daniel B. Rubin ^{4,5}, Ziv M. Williams ^{5,6}, Sydney S. Cash^{4,5}, Leigh R. Hochberg ^{4,5,7,8}, Lee E. Miller ^{3,9,10,11} & Chethan Pandarinath ^{1,12} 

Intracortical brain-computer interfaces (iBCIs) restore motor function to people with paralysis by translating brain activity into control signals for external devices. In current iBCIs, instabilities at the neural interface result in a degradation of decoding performance, which necessitates frequent supervised recalibration using new labeled data. One potential solution is to use the latent manifold structure that underlies neural population activity to facilitate a stable mapping between brain activity and behavior. Recent efforts using unsupervised approaches have improved iBCI stability using this principle; however, existing methods treat each time step as an independent sample and do not account for latent dynamics. Dynamics have been used to enable high-performance prediction of movement intention, and may also help improve stabilization. Here, we present a platform for Nonlinear Manifold Alignment with Dynamics (NoMAD), which stabilizes decoding using recurrent neural network models of dynamics. NoMAD uses unsupervised distribution alignment to update the mapping of nonstationary neural data to a consistent set of neural dynamics, thereby providing stable input to the decoder. In applications to data from monkey motor cortex collected during motor tasks, NoMAD enables accurate behavioral decoding with unparalleled stability over weeks-to months-long timescales without any supervised recalibration.

In people with paralysis, intracortical brain-computer interfaces (iBCIs) provide a pathway to restoring voluntary movements by interfacing directly with the brain to translate movement intention into action^{1,2}. iBCIs use implanted electrodes to record activity from populations of neurons and decoding algorithms to translate the recorded activity into control signals for external devices. In recent years, iBCIs have

attained impressive performance in a range of applications, including the control of anthropomorphic robotic arms, stimulation of paralyzed muscles to enable reaching and grasping, and even rapid decoding of handwriting^{3–6}.

Despite these impressive demonstrations, a key challenge limiting the clinical deployment of iBCIs is their robustness to neuronal

¹Wallace H. Coulter Department of Biomedical Engineering, Emory University and Georgia Institute of Technology, Atlanta, GA, USA. ²Center for Machine Learning, Georgia Institute of Technology, Atlanta, GA, USA. ³Department of Neuroscience, Northwestern University, Chicago, IL, USA. ⁴Department of Neurology, Massachusetts General Hospital, Boston, MA, USA. ⁵Harvard Medical School, Boston, MA, USA. ⁶Department of Neurosurgery, Massachusetts General Hospital, Boston, MA, USA. ⁷VA Center for Neurorestoration and Neurotechnology, Office of Research and Development, VA Providence Healthcare System, Providence, RI, USA. ⁸School of Engineering and Robert J. and Nancy D. Carney Institute for Brain Science, Brown University, Providence, RI, USA. ⁹Department of Biomedical Engineering, Northwestern University, Evanston, IL, USA. ¹⁰Department of Physical Medicine and Rehabilitation, Northwestern University, Chicago, IL, USA. ¹¹Shirley Ryan AbilityLab, Chicago, IL, USA. ¹²Department of Neurosurgery, Emory University, Atlanta, GA, USA.

 e-mail: chethan@gatech.edu

recording instabilities that cause changes in the particular neurons being monitored over time^{7–10}. Recording instabilities are attributed to a variety of phenomena, including shifts in electrode positions relative to the surrounding tissue, electrode malfunction, cell death, and physiological responses to foreign materials. As the particular neuronal population being monitored changes, so does the relationship between recorded neural signals and intention, which creates a non-stationary input to the iBCI's decoder. Without appropriate compensation, iBCI use must be periodically interrupted to perform supervised decoder recalibration, in which neural data are collected while subjects attempt pre-specified movements. This process can be required once or even multiple times per day to maintain high-performance², obstructing activities of daily living and creating additional burdens for iBCI users. Because the reliability of assistive devices is a key predictor of real-world use¹¹, iBCI instabilities are often cited as motivation for alternate neural interfaces such as electrocorticography, which offer more limited but potentially more stable performance¹².

Automatic, unsupervised decoder recalibration would provide a means to compensate for neural interface instabilities using only neural data collected during normal iBCI device use, thus preserving performance without interrupting use. One promising avenue to unsupervised recalibration is the use of latent, network-level properties of neural activity^{13–16}. In particular, a few recently developed iBCIs leverage latent manifolds, revealed by the patterns of co-activation within the neuronal population, as the foundation for a more stable neural interface^{9,17–20}. Manifold-based iBCI decoders use a two-stage approach: first, a *neurons-to-manifold* mapping that transforms recorded neuronal population activity onto the underlying manifold and second, a *manifold-to-behavior* mapping that transforms manifold activity into intended movements^{17–19,21}. Because manifolds are independent of the specific neurons being recorded, different sets of recorded neurons can be mapped onto the same manifold^{17–19,21–23}. And because these manifolds have a consistent relationship with behavior extending even to years^{21,23}, stable decoding can be achieved by properly recalibrating the neurons-to-manifold mapping without changing the manifold-to-behavior mapping.

A complementary avenue to improve the performance and stability of iBCI decoders is to incorporate latent dynamics, or the rules that govern the evolution of population activity over time²⁴. Models of neural population activity that incorporate dynamics have already shown promise for improving iBCI performance, as they produce representations that are informative of behavior on a moment-to-moment basis and millisecond timescale^{22,25–28}. Dynamics may also be useful for improving stability because dynamics, like manifolds, have a stable relationship with behavior for months to years and are independent of the specific population of neurons being monitored within a given area^{22,23}. To date, however, unsupervised efforts to stabilize iBCI decoding have not incorporated this temporal information.

Here we test Nonlinear Manifold Alignment with Dynamics (NoMAD), a platform for unsupervised stabilization of iBCI decoding. NoMAD uses a manifold-based iBCI decoder that incorporates a recurrent neural network model of dynamics. As instabilities cause changes in the recorded neural population, the learned dynamics model can be used to help update the neurons-to-manifold mapping without knowledge of the subject's behavior.

We applied NoMAD to recordings from monkey primary motor cortex (M1) collected during motor tasks in sessions that span multiple weeks and compared it to two previous state-of-the-art stabilization approaches that use latent manifolds. When applied to recordings from a monkey performing a two-dimensional isometric wrist force task, NoMAD achieved strikingly higher decoding performance compared to previous manifold alignment approaches without noticeable degradation over 3 months. Further, when applied to recordings from a behavior with very different output dynamics—a center-out reaching

task, with sessions spanning 5 weeks—NoMAD again achieved substantially higher decoding performance and stability than previous manifold alignment approaches. These results demonstrate that dynamics modeling can facilitate high-quality, unsupervised manifold alignment, greatly extending the period over which stable neural decoding is feasible and providing a pathway to more practical iBCIs.

Results

Leveraging manifolds and dynamics to stabilize iBCI decoding

We begin with a conceptual schematic (Fig. 1). As with previous manifold decoding approaches^{17–19,29,30}, our approach starts with a supervised training dataset containing neural activity and movement information from an initial recording session, which we call Day 0. We can use this dataset to characterize the manifold and dynamics, while also training a decoder to map manifold activity onto behavior. In some later recording period, termed Day K, neural recording instabilities have changed the specific neurons that are monitored, so electrode channels may now have a different relationship to the underlying manifold and dynamics. Thus, the original decoding axis no longer reflects the relationship between the manifold and behavior. As with other unsupervised methods, the high-level goal of NoMAD is to compensate for recording instabilities by learning a mapping from the Day K data onto the original manifold, allowing the original Day 0 decoder to be used. Unlike previous methods, NoMAD uses information about the temporal evolution of neural activity to help learn this mapping.

Adapting the LFADS architecture for manifold alignment using NoMAD

NoMAD models dynamics using latent factor analysis via dynamical systems (LFADS; Fig. 2a), a modification of standard sequential variational autoencoders (VAEs)^{31–33}, which has been previously detailed^{22,31,34}. Briefly, LFADS approximates the dynamical system underlying an observed neural population using a recurrent neural network (RNN; the “Generator”) that receives a sequence of inferred inputs. Additional RNNs encode the initial state of the dynamical system and infer the sequence of inputs to the Generator. The model outputs firing rate predictions for the observed neurons through a linear readout from the Generator—this readout sets the relationship between the learned manifold and the high-dimensional neural activity (i.e., it orients the manifold within the high-dimensional neural state space). In standard LFADS, the model training objective is to maximize a lower bound on the marginal likelihood of the observed spiking activity, given the firing rates output by the model. All weights in the model are updated during training via backpropagation through time.

To model the Day 0 supervised training dataset, we made two main modifications to the LFADS architecture. First, we added a low-dimensional read-in matrix at the model's input to standardize the dimensionality of the input to the RNNs. This read-in allows the same LFADS model to be applied to Day K datasets despite changes in the number of recorded electrodes from Day 0, such as those that may occur due to the removal of channels with sparse activity by the experimenter. Second, we added a readout matrix that predicts behavioral data from the Generator's activity, similar to methods that aim to learn dynamics that are related to both the recorded neural activity and measured behavioral variables^{35,36}. Behavioral prediction serves as a second training objective, adding a source of information that is complementary to the observed spiking activity^{35–37}. This helps to ensure that the model converges to a good solution using fixed hyperparameters, avoiding the need for computationally intensive automated hyperparameter tuning strategies such as AutoLFADS (Supplementary Fig. 1)²⁶. After training the LFADS model, we learn the manifold-to-behavior mapping, i.e., a decoder that predicts the Day 0 behavioral data from the Day 0 Generator states. Separating the training of the LFADS model and Day 0 decoder allows the use of

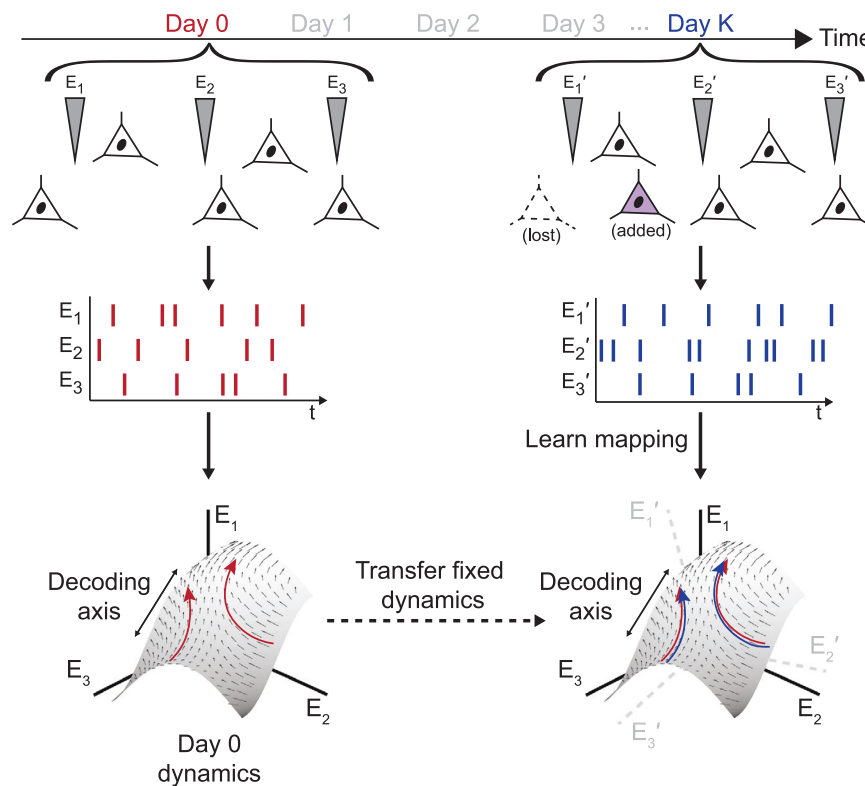


Fig. 1 | Manifold alignment with dynamics can stabilize representations of neural activity despite changes in recording conditions. We begin with a supervised training dataset containing neural activity and corresponding behavior from an initial recording session (*Day 0*). For a 3-electrode example (electrodes E_1, E_2, E_3), population activity exhibits an underlying manifold structure in a 3-D neural state space in which each axis corresponds to the firing rate from a given electrode. The evolution of population activity in time exhibits consistent dynamics (vector field). The relationship between manifold activity and behavior, for simple linear

decoding of a hypothetical 1-D behavioral variable, is represented by a *Decoding axis*, which is assumed to be consistent over time. In a subsequent recording session (*Day K*), instabilities lead to changes in the recorded neural population, and the *Day K* activity (E'_1, E'_2, E'_3) has a different relationship to the underlying manifold, dynamics, and decoding axis (schematized by a rotation). With NoMAD, our goal is to learn a mapping from the *Day K* neural activity to the original manifold and dynamics in an unsupervised manner. This allows the original decoding axis to be applied to accurately decode behavior.

decoding architectures with more complexity and capacity than a simple, single-time-step linear decoder (e.g., Wiener filters, RNNs, etc.) without impacting the learned manifold or dynamics.

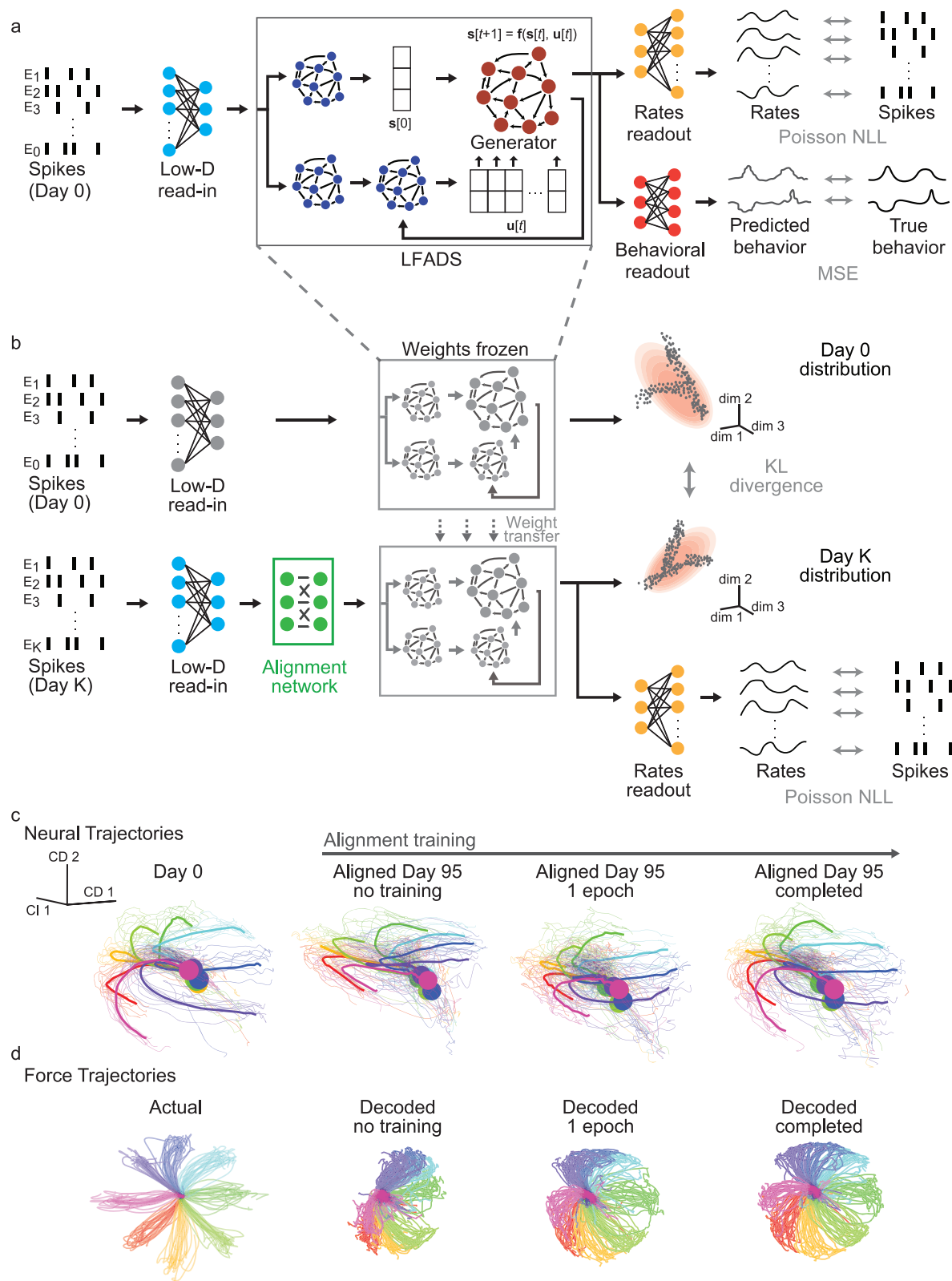
Because neural dynamics and behavior have a stable relationship over months to years^{22,23}, both the *Day 0* neural dynamics model and decoder should be applicable to data collected after the initial supervised dataset. However, because recording instabilities distort the relationship between recorded neurons and the manifold, we must periodically update the mapping between the neural activity and the manifold through an alignment transformation, which, fortunately, can be an unsupervised process. Once this transformation is learned, data on a subsequent *Day K* can be passed through the *Day 0* dynamics model, such that the *Day 0* decoder can be used to predict *Day K* behavior with high accuracy.

To update the *Day K* neurons-to-manifold mapping in NoMAD, the weights of the LFADS RNNs, including the Generator that expresses the latent dynamics, are held constant, while three other network components are learned or updated with an unsupervised alignment step: a feedforward alignment network that adjusts the input to the RNNs, the low-D read-in, and the rates readout (Fig. 2b). During the alignment step, there are two training objectives: (1) minimizing the difference between the distributions of the Generator states on *Day 0* and *Day K*, and (2) maximizing the likelihood of observed *Day K* spiking activity given the firing rates output by the model. At each training step, the *Day 0* and *Day K* Generator state distributions are compared by first approximating each by a multivariate normal distribution and then computing the Kullback-Leibler (KL) divergence between those normal distributions. The multivariate normal approximation focuses the

alignment process on matching first- and second-order statistical moments of the Generator state distributions, making the alignment problem more tractable than matching higher-order statistics. Model weights that are adjusted during alignment are updated via back-propagation through time.

While previous applications of LFADS have been acausal (i.e., data from the future is used to infer firing rates for a given timestep), here we perform inference using a sliding-window approach³⁸ to best simulate the usage of LFADS and NoMAD in a real-time, online iBCI environment (Supplementary Fig. 2). In this approach, the model's input consists of $N-1$ bins of previously collected data, and the N th time step is the most recent bin. This approach enables us to causally obtain predictions for one bin at a time with minimal latencies (see *Methods - Causal Inference*) and minimal changes in performance over acausal strategies (Supplementary Fig. 3).

To illustrate how NoMAD adjusts the manifold and dynamics, we applied it to twenty datasets collected from a monkey performing an isometric force task over 95 days. In this task, neural data were recorded from a 96-channel electrode array in M1 while a monkey generated wrist forces to control an on-screen cursor, with the goal of reaching a target in one of eight directions on each trial. We visualized the low-D structure of the Generator states by finding a 3-dimensional subspace via demixed principal component analysis (dPCA)³⁹. We first fit dPCA parameters to the Generator states inferred by the LFADS model from *Day 0* neural activity. We then applied those dPCA parameters to the Generator states on *Day K* to visualize three phases of unsupervised NoMAD alignment: before any training had occurred, after one training epoch, and after the completion of training (Fig. 2c).



By using the same dPCA parameters on Day 0 and Day K, we could directly compare the low-D trajectories across Day 0 and Day K. dPCA revealed a structure underlying Day 0 activity that distinguished neural activity from the 8 different target conditions. As an example quantification of the degree of alignment for each phase, we computed the Euclidean distance d between the condition-averaged dPCs on Day 0 and Day K. These measures are intended to provide intuition for how

the method works, and detailed quantification of alignment effectiveness is shown later in the manuscript. Before alignment, the low-D structure of the Generator states on Day K was distinct from the structure on Day 0, as expected ($d=16.2$). After alignment with NoMAD, the low-D structure on Day K more closely matched Day 0 ($d=12.8$ after one epoch and $d=7.13$ after training was completed). We then tested whether this unsupervised alignment led to greater

Fig. 2 | Modeling and aligning the neural manifold using NoMAD. **a** We first train an LFADS model on a given reference Day 0 to estimate the manifold and dynamics using a series of interconnected recurrent neural networks (RNNs). All of the model parameters are trained by simultaneously minimizing the reconstruction losses of the spiking activity (Poisson negative log-likelihood) and the recorded behavior (mean squared error). **b** To apply the same LFADS model on a later Day K, we freeze the parameters of the trained LFADS model and introduce a feedforward “Alignment network” to transform the new day’s spiking activity to be compatible with the previously trained LFADS model. The Alignment network is trained by simultaneously minimizing the KL divergence between the distributions of the Day 0 and Day K Generator states and the reconstruction loss of the spiking activity (Poisson

NLL). **c** Dimensionality reduction applied to the Generator states for Day 0 and throughout alignment training for data collected 95 days later. Neural data were recorded from M1 as a monkey performed an isometric force task. Colors indicate trajectories for the different target locations. Thick lines indicate the trial average, and thin lines denote single trials (10 representative trials per condition are shown here). Initial estimates of Day 95 Generator states converged over training to resemble Day 0 Generator states, allowing the same decoder to achieve high-accuracy force predictions. **d** Left: Measured single trial force trajectories, colored by target location. Right: decoded forces across several epochs of Day 95 alignment to Day 0: before alignment ($R^2 = 0.59$), after 1 epoch of alignment training ($R^2 = 0.80$), and after completed alignment training (116 epochs; $R^2 = 0.88$).

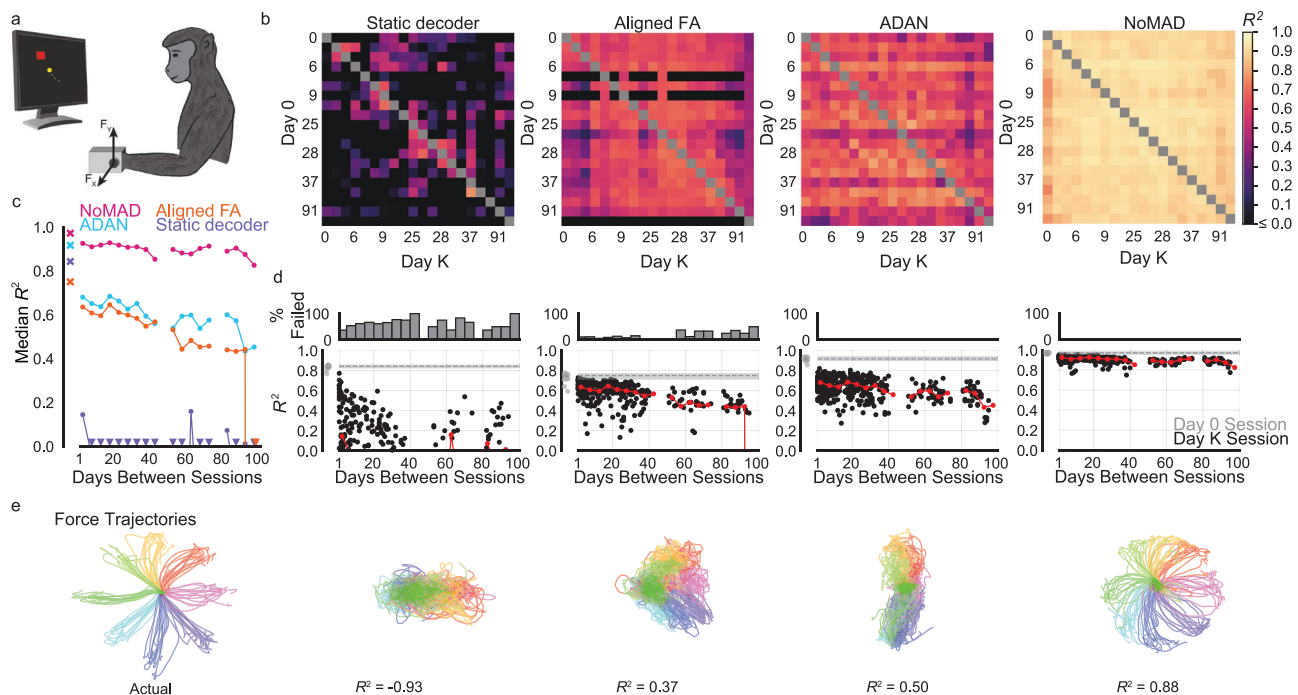


Fig. 3 | NoMAD enables stable offline decoding across months in an isometric force task. **a** Schematic of isometric force task, modified from figure in Ethier, et al.⁶⁸. **b** R^2 of aligned force decoding for each method applied to all pairs of sessions (20 sessions spanning 95 days). **c** Median R^2 of force decoding within each 5-day bin. Triangles indicate points that fall below the limit of the y-axis and x-markers indicate median within-day force decoding performance on Day 0. **d** Top: Percentage of Day K decoding failures ($R^2 < 0$) in 5-day bins. Bottom: Decoding performance as a function of days between sessions. Black points

indicate Day K decoding performance for single pairs of days. Red points show the median Day K performance within each bin of width 5 days. Gray points denote the initial performance for Day 0 decoders (i.e., evaluated on held-out same-day data) for each of the 20 datasets. Gray dashed lines indicate the median decoding performance of Day 0 decoders, and shaded regions around them represent the first and third quartiles. **e** Left: Measured single-trial force trajectories on Day 95, colored by target location. Right: decoded forces for all methods when Day 95 is aligned to Day 0.

decoding stability by first training a Wiener filter decoder that mapped the Generator states to the measured forces on Day 0, and then testing this decoder on Day K data over several epochs of NoMAD alignment. As shown, the alignment procedure improved force decoding substantially (Fig. 2d).

NoMAD stabilizes offline decoding during an isometric force task

We applied NoMAD to pairs of recording sessions from a monkey performing the isometric force task over 95 days (Fig. 3a). Behavioral decoding consisted of a Wiener filter that mapped manifold activity onto the recorded 2-D forces with high temporal resolution (20 ms time steps). The Day 0 dynamics model (LFADS) and Day 0 decoder (Wiener filter) were trained on a supervised dataset from a single session, and we evaluated NoMAD’s ability to enable accurate and stable decoding performance on a different session (Day K) through unsupervised alignment. We compared NoMAD against a standard

Wiener filter decoder that was trained using smoothed spiking activity and behavior from Day 0 and evaluated on Day K without any adjustment (*Static decoder*), and also to two state-of-the-art manifold-based stabilization techniques: aligned factor analysis (Aligned FA)¹⁷, which is based on linear dimensionality reduction, and the adversarial domain adaptation network (ADAN)^{18,19}, which uses a neural network auto-encoder for dimensionality reduction and generative adversarial networks for alignment. We use the same Wiener filter decoding strategy for all methods (Supplementary Fig. 4). We note that NoMAD alignment is completely unsupervised, in that we use all available data from a session, including periods where the monkey was inactive. For Aligned FA and ADAN, we use only within-trial data where the monkey is active, to be consistent with the original demonstrations of these methods (explored further for Aligned FA in Supplementary Fig. 5). We quantified Day K force decoding using the coefficient of determination (R^2), i.e., the fraction of variance of the recorded force signal on Day K that is predicted by the decoder. We term evaluations with negative R^2

to be decoding failures. We quantify stability using half-life by converting the median Day K R^2 values to signal-to-noise ratio (SNR) and fitting an exponential decay curve (Supplementary Fig. 6) as well as by quantifying the drop in R^2 for pairs of sessions separated by one day (Supplementary Fig. 7). In our tests, across-days applications of alternative dynamical models (e.g., Targeted Neural Dynamical Modeling³⁶) did not achieve comparable stability to the NoMAD approach (Supplementary Fig. 8).

Before evaluating across-session decoding stability, we first evaluated baseline decoding performance for each method by training Day 0 decoders for each session and testing those decoders on held-out data from the same session. Day 0 decoders used for NoMAD, which were trained on Day 0 LFADS Generator states, achieved the highest median performance and had the least amount of variability for the twenty sessions (median R^2 [Q1, Q3] = 0.971 [0.965, 0.974]), followed by ADAN (0.916 [0.908, 0.929]). Aligned FA (0.749 [0.722, 0.761]) and the Static decoders (0.842 [0.834, 0.846]) had lower and more variable within-day decoding performance. These initial Day 0 performance metrics serve as a useful upper bound in interpreting the efficacy of each alignment method.

To measure decoding stability across a wide variety of recording conditions, we tested each method on all pairs of sessions (380 pairs; Fig. 3b). As expected, Static decoders trained on a given session failed to generalize to other sessions. This resulted in rapid degradation of decoding performance over time and frequent decoding failures, such that the median decoding performance was quite poor for pairs of sessions spaced less than 5 days apart ($R^2 = 0.14$) and negative for pairs spaced further apart (223 decoding failures; Fig. 3c, d).

Aligned FA achieved more stable decoding than the Static decoder across time (half-life = 45.1 days) but with only moderate performance (median $R^2 = 0.59$; $p = 1.76 \times 10^{-19}$, one-sided Wilcoxon signed-rank test between force decoding R^2 values for all pairs of sessions) and high variability. In addition, Aligned FA repeatedly failed for most

alignments that were initialized using particular sessions (shown by the black horizontal bars in the heatmap), resulting in 51 total decoding failures. Separately, we also attempted to apply Aligned FA sequentially, i.e., performing alignment between sequential recording sessions to try to maintain stability (the approach described in the original study), but found that this further decreased performance and increased variability (Supplementary Fig. 9). ADAN improved performance further relative to the Static decoder and Aligned FA (median $R^2 = 0.65$; $p = 3.18 \times 10^{-64}$ and 3.96×10^{-30} , respectively, one-sided Wilcoxon signed-rank test), and also improved stability (half-life = 76.7 days), with 0 decoding failures. Yet, performance for individual pairs of sessions was highly variable.

Compared to these previous methods, NoMAD achieved strikingly higher performance (median $R^2 = 0.91$; $p = 2.53 \times 10^{-64}$ for all of ADAN, Aligned FA, and Static decoder, one-sided Wilcoxon signed-rank test) with little variability, no failures, and minimal performance degradation across the 3-month window (half-life = 208.7 days). These differences were also evident when visualizing the decoded data: single-trial forces decoded by NoMAD were more consistent with the measured forces than those produced with other methods (Fig. 3e).

NoMAD stabilizes offline decoding during an unloaded reaching task

To ensure that NoMAD's efficacy extends beyond the isometric task, we also evaluated its performance on recording sessions from a monkey performing a center-out reaching task^{21,40}. On each trial, a monkey used a manipulandum to move a cursor from the center of a screen to one of eight targets spaced equally around a ring (Fig. 4a). We applied NoMAD to 96 channels of data recorded from M1 over 12 sessions spanning 38 days. As in the isometric task, we used a supervised dataset to train the Day 0 LFADS model and Wiener filter decoder. We evaluated NoMAD on a separate Day K dataset by performing unsupervised alignment before applying the Day 0 decoder.

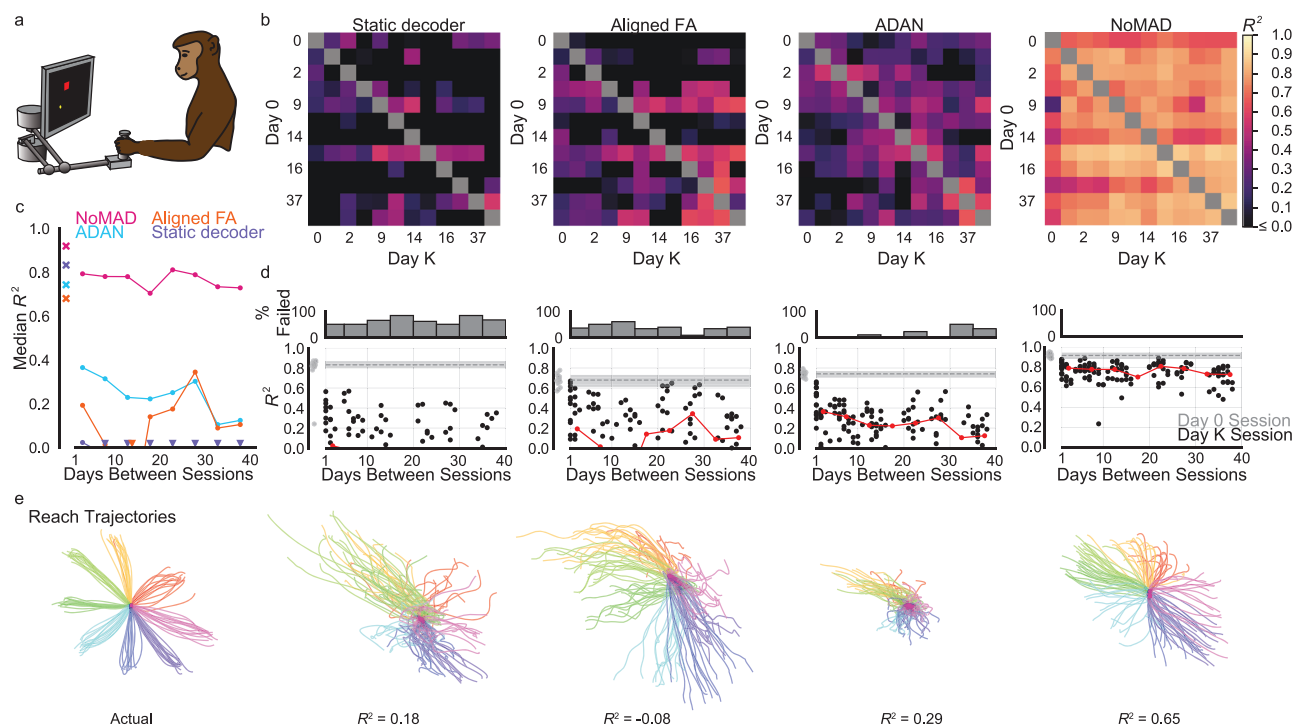


Fig. 4 | NoMAD enables stable offline decoding across 5 weeks in a reaching task. **a** Schematic of the center-out reaching task, modified from figures in Ethier et al.⁶⁸ and Perich et al.⁴⁰ Copyright (2018), with permission from Elsevier. **b** R^2 of aligned cursor velocity decoding for each method applied to all pairs of sessions (12 sessions spanning 38 days). **c** Median R^2 of cursor velocity within each

5-day bin. Conventions as in Fig. 3. **d** Top: Percentage of Day K decoding failures ($R^2 < 0$) in 5-day bins. Bottom: Decoding performance as a function of days between sessions. Conventions as in Fig. 3. **e** Left: Measured single-trial reach trajectories, colored by target location. Right: reach position trajectories integrated from the decoded reach velocity when Day 38 is aligned to Day 0.

We again tested NoMAD's performance against ADAN, Aligned FA, and a Static decoder on the same data^{37–39}. Day K decoding was quantified using the R^2 between the recorded and predicted cursor velocity signals, and negative R^2 values were again classified as decoding failures.

We again started by evaluating baseline decoding performance for all sessions using decoders trained on Day 0 data and evaluated on held-out Day 0 data for each method. NoMAD achieved the highest and least variable performance amongst the twelve sessions (median R^2 [Q1, Q3] = 0.918 [0.896, 0.934]). Static decoders fit to smoothed spikes were the next highest performing for this dataset, but were highly variable (0.832 [0.813, 0.856]). ADAN within-day decoders' performance was lower but had variability similar to NoMAD (0.742 [0.716, 0.757]). Aligned FA yielded initial decoders with the lowest performance and highest variability on this task (0.679 [0.624, 0.718]).

We next assessed whether each method could facilitate stable decoding across sessions by testing alignment on all pairs of days (132 pairs; Fig. 4b). Static decoders again failed to yield accurate decoding across sessions, with poor median performance even for sessions separated by less than five days ($R^2 = 0.022$), and primarily negative performance thereafter. A total of 78 pairs of days result in decoder failures (Fig. 4c, d).

This dataset was more challenging to align for Aligned FA and ADAN than was the isometric task. Aligned FA provided insignificant accuracy improvement over the Static decoder (median $R^2 = 0.17$; $p = 0.153$, one-sided Wilcoxon signed-rank test between cursor velocity decoding R^2 values for all pairs of sessions). Again, there was significant variation in decoding performance, which resulted in 53 decoding failures and a clear decline in performance (half-life = 1.03 days). ADAN provided some improvement in terms of accuracy (median $R^2 = 0.29$) and stability (half-life = 7.12 days, 15 decoding failures) but still showed high variability amongst performance for individual pairs ($p = 5.84 \times 10^{-6}$, 1.53e-17 compared to Aligned FA and Static decoder, respectively, one-sided Wilcoxon signed-rank test). The differences in performance on this reaching task when compared to the isometric task may result from an increased complexity of behavior, requiring different coordinated patterns of muscle activity, or from inter-subject differences in array properties at the time the data were recorded.

NoMAD exhibited higher decoding performance (median $R^2 = 0.78$) with no decoding failures and less variability in performance over the 5-week timespan ($p = 1.07 \times 10^{-23}$, 1.07e-23, 1.05e-23 for ADAN, Aligned FA, Static decoder, one-sided Wilcoxon signed-rank test). NoMAD also showed less degradation in decoding accuracy (half-life = 57.9 days). Visualizations of the decoded cursor trajectories confirmed that NoMAD produced more consistent cursor velocity estimates following stabilization than other methods (Fig. 4e).

NoMAD is complementary to retrospective decoder recalibration approaches

Manifold-alignment approaches are completely unsupervised and use only neural data to perform a recalibration. As demonstrated in Figs. 3, 4, manifold alignment performed with NoMAD can be effective on timescales of days to months.

Another class of approaches attempts to recalibrate decoders based on retrospective analysis of the subject's use of the iBCI^{41,42}. In particular, retrospective target inference (RTI)⁴¹, a leading method, operates in settings where the user's intent can be guessed post hoc, as in BCI spellers that use a limited number of predefined targets. In these settings, neural activity preceding movement to a known target can be assumed to reflect the user's intention to move toward that target. These data can then be used to update the decoder after the fact, much like supervised decoder training. A recent advance on this method enables estimation of the user's intent during more complex behaviors using a probabilistic model and may lessen the amount of task information necessary for retrospective decoder recalibration⁴². Retrospective decoder recalibration approaches have been demonstrated to

be highly effective when applied within sessions of closed-loop iBCI control over timescales of minutes to hours^{41,42}.

While manifold alignment uses only neural activity to stabilize manifold representations, retrospective decoder recalibration focuses on revising the relationship between the neural activity and the behavior (captured by the decoder). Because these classes of approaches operate on separate components of the decoding pipeline, NoMAD and RTI could potentially be used together in a complementary manner. Here, we wanted to determine whether it was possible to combine manifold alignment approaches (NoMAD) with retrospective decoder recalibration approaches (RTI), and assess how this combination of approaches affected decoding stability over a short timescale.

We analyzed closed-loop human iBCI data from T11, a participant in the BrainGate2 pilot clinical trial. On each trial, T11 was asked to imagine moving his right hand to control the cursor from the center of the screen to one of four outer targets. We defined the intended velocity of each successful trial as a constant speed over a 500 ms time period and used this as the behavioral signal to train the decoder. The decoder used was an optimal linear estimator for all approaches. We compared the stability and accuracy of NoMAD + RTI to NoMAD alone, RTI alone, and a Static decoder baseline. To simulate how these approaches may be applied during a session of iBCI use, each recalibration approach was applied on a per-block basis, where recalibration incorporates all available blocks preceding the evaluation block in question (see Supplementary Fig. 10). We applied this procedure separately for two sessions (Fig. 5b, c).

We first assessed how well the baseline decoders performed for each method. The Static decoder trained on smoothed spikes achieved predicted velocity $R^2 = 0.47$ and $R^2 = 0.60$ for each of the two tested sessions, respectively. This decoder was the calibration block performance for the Static decoder as well as the RTI decoder because no recalibration data had yet been collected. The decoders trained on LFADS Generator states achieved much higher performance at $R^2 = 0.86$, 0.86 for each session. Similarly, this decoder served as the starting point for both NoMAD and NoMAD + RTI, as no recalibration or alignment had occurred.

These decoders were kept unmodified for the first evaluation block because no additional historical data was available for recalibration or alignment. For the first session, the NoMAD and NoMAD + RTI decoder dropped to $R^2 = 0.73$, and the Static and RTI decoder dropped to $R^2 = 0.36$. For the second session, the NoMAD and NoMAD + RTI decoder resulted in $R^2 = 0.81$, while the Static and RTI decoder gave $R^2 = 0.53$.

We considered all remaining blocks to be Alignment/Recalibration blocks; the decoding performance for each of these blocks was reported after NoMAD was used to apply unsupervised alignment and/or RTI was used to recalibrate the decoder with additional trials after retrospectively inferring T11's intent. For both sessions tested, the Static decoder achieved the lowest decoding performance over this time period (median $R^2 = 0.32$ [0.23, 0.37] and 0.49 [0.45, 0.51]), with variable stability for each session (half-life = 3.26 min, 7.02 days).

RTI improved decoding performance over the Static decoder (0.46 [0.41, 0.49], 0.67 [0.63, 0.70]; $p = 0.016/0.0039$, for session 1/2, one-sided Wilcoxon signed-rank test between velocity decoding R^2 values). Its performance decayed quickly for the first session (half-life = 3.21 min), while for the second session, performance increased over time (doubling time = 2.40 h). Applying manifold alignment using NoMAD resulted in higher decoding performance for both sessions (0.66 [0.63, 0.70], 0.79 [0.76, 0.83]; $p = 0.016/0.0039$, 0.016/0.0039 for Static decoder, RTI in one-sided Wilcoxon signed-rank test) with relatively high stability within each session (half-life = 5.60 h, 21.05 h). NoMAD's stability exceeded that of RTI for the first session, while RTI's stability exceeded that of NoMAD for the second session.

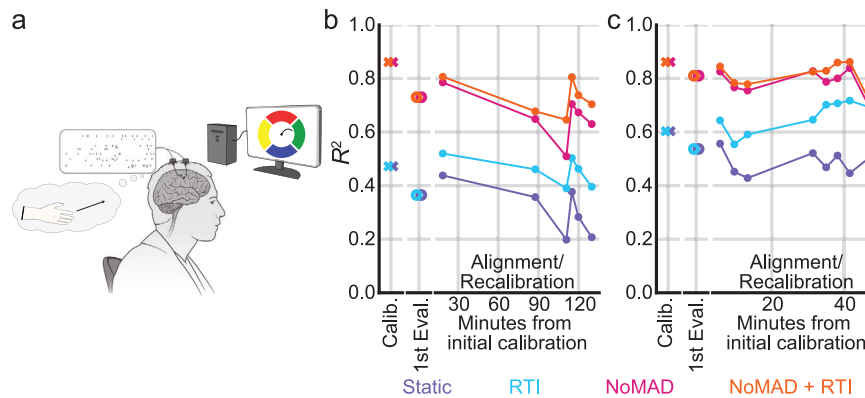


Fig. 5 | Demonstration of NoMAD and retrospective target inference (RTI) on closed-loop iBCI data. **a** Schematic of the center-out cursor control task, modified from figure in Willett et al.⁶. On any given trial, the participant controlled the cursor from the center to one of the outer four targets. Upon successful target acquisition, the cursor was automatically returned to the center. **b**, **c** R^2 of predicted cursor velocity compared to estimated intended cursor velocity for each of the methods

during two sessions. Each initial calibration block's decoding performance is denoted with an x. The first evaluation block, during which no recalibration has yet taken place, has decoding performance denoted by an open circle marker. Decoding performance after recalibration and/or alignment has been performed using all previous blocks is denoted using closed circle markers.

Applying NoMAD in combination with RTI gives the highest decoding performance of all tested methods (0.72 [0.68, 0.79], 0.83 [0.78, 0.85]; $p = 0.016/0.0039$, $0.016/0.0039$, $0.016/0.0078$ for Static decoder, RTI, NoMAD in one-sided Wilcoxon signed-rank test). NoMAD + RTI also achieved consistently stable performance within each session (half-life = 11.73 h, 4.47 days). For one session, these results indicate that NoMAD + RTI provided higher performance and stability over each approach applied independently; for the second session, NoMAD + RTI provided reasonable, though not superior, performance and stability.

NoMAD + RTI was able to achieve performance and stability that is comparable to each approach independently when applied to closed-loop iBCI data over short timescales. These results demonstrate that it is feasible to combine manifold alignment and retrospective recalibration approaches without worsening performance or stability. The effect of this combination varied across the tested sessions; further study may be necessary to determine under which conditions manifold alignment combined with retrospective target inference is most successful.

Discussion

We introduced NoMAD, an unsupervised manifold alignment technique that leverages manifolds and their dynamics to achieve stable decoding over long timespans. In our tests, NoMAD improved both decoding accuracy and stability over 95 days in an isometric wrist task and over 38 days in a reaching task. Other manifold alignment methods resulted in less decoding improvement or marked performance degradation over time. By incorporating the temporal structure of the neural activity into the manifold-alignment process via dynamics, NoMAD enables more stable neural decoding that may require less frequent iBCI recalibration procedures. Short-timescale stability may also be improved by combining NoMAD with retrospective decoder recalibration approaches.

Relation to previous work

Our method improves upon previous manifold alignment efforts by incorporating temporal constraints via nonlinear dynamics models. In addition, the method is entirely unsupervised, in contrast to methods such as canonical correlation analysis (CCA) and previous methods that exploit dynamics to improve iBCI longevity^{21,23}. A variety of alternate strategies have been used to reduce the reliance on supervised decoder recalibration^{10,17–19,29,30,41}. One approach uses neural

network decoders and months-long datasets that expose the decoder to a wide variety of recording instabilities to learn a mapping from neuronal population activity to movement intention that is robust to changes in the recorded neurons¹⁰. However, collecting such large supervised datasets requires a substantial time commitment from the user and is therefore challenging to perform clinically. Other methods may rely on models of neuronal tuning to perform automatic decoder recalibration, as in Bayesian regression self-training⁴³, but these methods make strict assumptions about neural computations that may not hold in all brain regions and BCI usage scenarios. NoMAD inherently has no dependence on the nature of the behavior, which may allow NoMAD alignment to be applied to BCI usage scenarios where the behavioral intention is harder to estimate. Nonetheless, supervised or intention estimation strategies may provide a complementary approach when combined with manifold alignment.

Another manifold alignment approach, Distribution Alignment Decoding (DAD), is conceptually similar to the approach used here but performs alignment of low-dimensional activity from two datasets by doing a brute-force search of candidate rotations in the low-D space²⁹. However, this alignment approach fails when distributions of movements are symmetric. Further, when dimensions of neural activity scale beyond extremely simple representations (e.g., beyond 2 or 3 dimensions), a brute-force search quickly becomes intractable. More recently, Hierarchical Wasserstein Alignment (HiWA) has improved on this approach using neural networks³⁰, but it relies on the existence of a discrete structure, such as clusters within the neural activity that represent similar movements.

LFADS has also been used previously to identify manifolds and dynamics that are shared across multiple datasets recorded on separate days, a technique known as “stitching”²². Stitching serves as a demonstration that multi-day neural dynamics models are feasible. However, stitching with LFADS is a supervised approach, requiring model components to be specifically initialized using behavioral information from all datasets to encourage the learning of a shared subspace; our attempts to adapt LFADS stitching to perform this task in an unsupervised manner compatible with iBCI use have been unsuccessful (Supplementary Fig. 11). These results indicate that without some sort of alignment pressure, models such as LFADS may find different solutions – effectively different manifolds or dynamics – to model different recording sessions, preventing decoder transfer between sessions. As such, our use of LFADS for dynamics-based manifold alignment includes additional model components (e.g., the

alignment network) and training objectives (e.g., KL cost) that are necessary to robustly improve unsupervised iBCI recalibration (Supplementary Fig.12).

Our study makes several parameter choices that are relevant for closed-loop iBCI applications but may affect reported model performance. Within each method, we used the same hyperparameters for both monkey tasks, which required the model to be robust to these choices. With more task-specific tuning, model performance may modestly improve. In addition, our study operated on bin size parameters selected to be appropriate for closed-loop BCI applications. Large bin sizes could cause long latencies and degrade closed-loop iBCI performance⁴⁴. Previous works, including those that introduced ADAN and Aligned FA, used larger bin sizes than are demonstrated here, but which would incur latencies inappropriate for iBCI use. NoMAD's dynamical-systems-based approach allows larger time windows to be taken into account during modeling while still ensuring high-accuracy decoding at low latencies.

We chose to perform alignment with respect to a single reference session regardless of the time elapsed between sessions. In our tests, this performed better than aligning sequentially (Supplementary Fig. 9) and has the additional benefit of enabling testing of a variety of alignment conditions and timescales. Aligning to a single reference session also simplifies the approach by avoiding error aggregation that may occur due to repeated alignment to a method's output. In such cases, it is not clear how a method should handle any reductions in alignment quality that occur; aligning to a well-validated Day 0 manifold estimate skirts this issue entirely. Future applications of alignment approaches should evaluate which alignment scheme yields the highest empirical performance for their specific scenarios.

Our work aims to evaluate the stability of the representations produced by each alignment approach by using decoding performance as an iBCI-relevant metric. To this end, we selected a simple linear decoding method to maintain focus on the stability of the aligned representations and their ability to yield consistent behavioral predictions. The practice of using linear methods to probe representation quality is well-established in the representation learning community⁴⁵. A key challenge of assessing representations output by a machine learning model – which distinguishes this task from other machine learning tasks – is the difficulty of establishing a clear performance objective. Thus, efforts to assess representations focus on linear readouts (i.e., decoders), which provide the ability to probe representation quality while making as few assumptions about the performance objective as possible. Nonlinear decoders raise several additional questions regarding hyperparameter optimization for manifold representations with varying levels of noise, such as optimal learning rates, regularization, and training schedules. Recent work has also indicated that nonlinear decoders can overfit in ways that degrade online iBCI control, making them more challenging to employ and interpret for our intended application⁴⁶.

A practical limitation of the approach is that the NoMAD alignment process, like most neural network training, is computationally demanding. Thus, without further optimization for implanted medical devices with severe power constraints, the alignment process is best run on external computing hardware. Further improvements to NoMAD's computational efficiency, such as precomputing the parameters of the Day 0 distribution to eliminate the need to store and process Day 0 data, may additionally improve its compatibility with wireless device development. Despite such constraints, the benefits of NoMAD to stability and accuracy indicate a promising advance to iBCIs.

Potential future applications

In this work, we test the NoMAD approach with particularly rigorous constraints, specifically: (1) the alignment process must be completely unsupervised (i.e., cannot incorporate any behavioral information),

and (2) only a small (minutes-long) initial supervised training dataset is used to calibrate the model. Even with these requirements, we found that NoMAD could stabilize decoding performance for many weeks to months. However, these constraints could be relaxed in a practical application. For example, assuming periods of reasonable decoding stability, data collected during BCI use comes with an inherent set of behavioral labels (i.e., we know how the subject was using the BCI), and this information could be incorporated to guide the alignment process. Similarly, as more and more datasets are collected during BCI use for a given subject, one could train an aggregated model that spans those datasets (similar to previous work^{10,22}), which would provide decoders that become inherently more stable as more data is collected, thus lessening the need for frequent recalibration.

NoMAD's performance demonstrates that the use of dynamics to infer the neural manifold can improve the accuracy and stability of manifold alignment procedures. While NoMAD's alignment procedure relies on a feedforward network trained to minimize KL Divergence, the NoMAD framework could be made compatible with several alignment approaches, such as adversarial networks or cycle costs, that could further improve performance.

We note that while the current work focuses on spiking activity recorded via intracortical electrode arrays, this is not an inherent limitation of the approach. Indeed, LFADS, upon which NoMAD is based, has been applied successfully to improve behavioral state classification from electrocorticographic (ECoG) recordings⁴⁷, which suggests that the NoMAD approach could be made to generalize to other BCI recording modalities and signal sources. Previous literature has also suggested that signal sources such as local field potentials (LFP) may provide more decoding stability than spikes⁴⁸. Taken in combination with latent variable models and unsupervised alignment approaches such as NoMAD, these signals may be used to further enhance decoding stability.

Recent efforts have been focused on improving the inherent stability of electrode arrays, for example, through biomaterials and tissue engineering innovations that reduce the inflammatory effect on surrounding tissue, promote cell growth, and enable flexible movement with the brain^{49–54}. However, with electrode arrays currently approved for chronic implantation in human subjects, clinical iBCIs remain subject to neural interface instabilities that necessitate algorithmic solutions². As new neural interfaces progress to clinical applications, they are likely to complement algorithmic stabilization approaches, resulting in improved stability beyond what is possible with either approach individually.

A potential limitation of current manifold alignment methods is that they rely on a stable relationship between activity on the neural manifold and behavior over time. While this assumption is reasonable for previously learned behaviors^{21,22}, it may not be the case if subjects are learning new skills. Indeed, though learning need not change the manifold on short timescales⁵⁵, long-term learning likely results in manifold-level changes⁵⁶, which might also affect the relationship between manifold activity and behavior. However, as demonstrated by several methods including ours^{17–19,29,30}, stabilization does not require massive data libraries (e.g., spanning many months), but instead can be achieved using recording sessions lasting only tens of minutes. Therefore, if the latent manifold and dynamics are changing due to processes like learning, stabilization could be run at shorter temporal intervals over which the manifold and dynamics should be locally stable. Further studies in this area could help determine whether and how manifold-alignment techniques need to account for learning.

Another limitation of the tests of manifold stabilization approaches to date is the consistency of behavior in the datasets used. The behavioral consistency achieved in the lab setting ensured that the datasets were rich enough and similar enough to be alignable. In practical applications, a key assumption of all stabilization approaches is that iBCI decoding performance will be stable for certain time

periods, such that data can be collected during use of the iBCI and periodic alignment can be performed to maintain manifold stability. If decoding performance is stable for reasonable time periods without alignment (e.g., many hours), this could ensure that the datasets cover rich enough and similar enough behavioral distributions (e.g., by spanning large enough regions of behavioral space) for successful periodic alignment.

In addition, over extremely long timescales (e.g., many months to years) that exceed those tested in this work, the likelihood increases that this method's assumptions of a stable manifold and consistent behavior will be violated. Future studies may work to characterize the efficacy of manifold alignment approaches over such time spans and establish the reasonable limit during which these approaches can be reliably applied.

An open neuroscientific question is the degree of similarity in manifold structure and dynamics across behaviors. This question has profound implications for building BCIs that generalize across behaviors. Recent studies suggest that different behaviors may occupy distinct manifolds⁵⁷. As such, BCIs that rely on manifold structure may require different mappings from manifold activity to behavior and would need to adjust their decoding depending on the manifold or manifolds that are currently occupied. This also affects stabilization strategies—to date, manifold-based stabilization methods have been tested only on datasets containing single behaviors. However, solutions to address the multiple-manifold scenario exist, including labeling data collected during BCI use by the behavior that was being performed and using that information to guide alignment.

Our offline measures of NoMAD's performance demonstrate the potential of neural population dynamics to yield long-term stability and accuracy for iBCI decoders. Offline experiments can be a useful way to evaluate alignment approaches as they remove the confound of potential user compensation for any decoder shortcomings. Nevertheless, we acknowledge that offline demonstrations of alignment approaches may not be representative of results in an online setting due to differences in neural dynamics, user strategy, amount of available training data, and necessary frequency of unsupervised recalibration. Prior literature has established that maximizing offline decoding accuracy does not necessarily lead to improvements in online performance, both for traditional linear models^{25,44,58–60} and neural networks^{10,46} (reviewed in Pandarinath and Bensmaia, *Physiol. Rev.* ²). Future work will prioritize demonstrating NoMAD in online iBCI experiments, bolstered by the promise of previous demonstrations of dynamical systems models in such contexts; experiments using LFADS, the underlying dynamical systems model comprising NoMAD, have demonstrated its feasibility for use in an online iBCI setting with sub-10ms latencies (Supplementary Fig. 13) and minimal decreases in performance⁶¹. The application of latent variable models to online iBCI control is still in its infancy, and future investigation into the benefits of NoMAD as an unsupervised recalibration procedure during online iBCI use will further illuminate how it may impact device development and whether it may lead to more feasible real-world devices.

Methods

Monkey surgical implants

After training animals to perform the tasks, a 96-channel microelectrode array with 1.5 mm-long electrode shanks (Blackrock Microsystems, Salt Lake City, Utah) was implanted into the hand area of the primary motor cortex (M1). Prior to implanting the array, the hand area of M1 was identified intraoperatively through sulcal landmarks and by stimulating the surface of the cortex to elicit twitches of the wrist and hand muscles.

Monkey data collection

Collection. Cortical data was collected using the Cerebus data acquisition system from Blackrock Microsystems. Raw cortical data was

collected at 30 kHz and filtered with a 250 Hz high-pass filter. Time-stamps of when the filtered data passed below a provided threshold were recorded as threshold crossings, commonly also called spikes. The threshold used for spike detection was set according to the root-mean square (RMS) activity on each channel and kept consistent across different recording sessions; for both monkeys, the threshold was set to $-5.5 \times \text{RMS}$. This process was all done using software on the Cerebus system.

Force data for the Isometric Task was collected at 2 kHz using the Cerebus analog inputs. Manipulandum joint data for the Reaching Task was collected using a NI DAQ card connected to a Mathworks XPC system that controlled the task. The XPC system calculated the end-point coordinates of the manipulandum, then sent the coordinates to the Cerebus through a series of digital packets.

Binning. The data was binned into 1 ms segments. For the threshold crossings, the number of spikes per bin was counted. For the force and manipulandum position data, the data was filtered and downsampled to 1 kHz. The cortical and task bins were aligned in time.

Monkey isometric task

Monkey J was trained to operate a 2D isometric wrist force device. The monkey's left arm was positioned in a splint to immobilize the forearm in an orientation midway between supination and pronation (with the thumb upwards). A small box was placed around the monkey's open left hand, incorporating a six-axis load cell aligned with the wrist joint. The box was padded to comfortably constrain the monkey's hand and minimize its movement within the box. The monkey controlled the position of a cursor displayed on a monitor by the force exerted on the box. Flexion and extension forces moved the cursor right and left, respectively, while forces along the radial and ulnar deviation axis moved the cursor up and down. Prior to placing the monkey's hand in the box, the force was nulled in order to place the cursor in the center target. Targets were displayed either at the center of the screen (zero force), or equally-spaced along a ring around the center target.

The monkey performed a center-out task that began with the appearance of the center target. The monkey was allowed two seconds to move to the center target and was required to hold for a time randomly chosen from a uniform distribution between 0.2 s and 1.0 s. A successful center hold triggered the appearance of one of eight possible outer targets, chosen in a block-randomized fashion. The monkey was allowed another two seconds after target onset to move the cursor to the outer target. The required hold time for the outer target was 0.8 s. Successful trials ended with the delivery of a liquid reward. Failure to reach a target within the allowed two seconds or to remain within a target as required resulted in an aborted (center target) or failed (outer target) trial. Successive trials were separated by a two-second inter-trial interval. This data has been published in Ma*, Rizzoglio*, et al., *eLife* 2023.

Monkey reaching task

Monkey C was trained to make reaching movements using a planar manipulandum in a two-dimensional center-out manner. To begin each trial, the monkey moved his hand to the center of the workspace. After a waiting period, the monkey was presented with one of eight equally spaced outer targets, arranged in a circle and selected uniformly at random. The monkey was trained to hold for a variable delay period, after which he received an auditory go cue. After the go cue, the monkey had 1 s to reach the outer target and hold within the target for 0.5 s. A successful trial led to a liquid reward. The cursor position was recorded at 1 kHz using joint encoders. Timing events, such as go cues, were logged digitally. This data was previously published in Perich et al., *Neuron* 2018; Gallego, et al., *Nature Neuroscience* 2020; and Ma*, Rizzoglio*, et al., *eLife* 2023.

All surgical and experimental procedures were approved by the Institutional Animal Care and Use Committee (IACUC) of Northwestern University under protocol #IS00000367 and are consistent with the Guide for the Care and Use of Laboratory Animals.

Human participant details

Participant T11 is a 36-year-old right-handed man with a history of tetraplegia secondary to a cervical spinal cord injury. He had two 96-channel intracortical microelectrode arrays (Blackrock Microsystems, Salt Lake City, Utah) placed chronically into the left precentral gyrus (PCG) as part of the BrainGate pilot clinical trial (www.ClinicalTrials.gov; Identifier: NCT00912041), with placement occurring 153 days and 427 days prior to the recording sessions. In the time periods before, between, and after the research sessions used in this paper, the participant has taken part in twice-weekly research sessions for other studies and had frequent practice using the BrainGate neural interface system to control a computer cursor. All research sessions were performed at the participant's place of residence. Permission for this study was granted by the U.S. FDA (Investigational Device Exemption #G090003) and the IRBs of Massachusetts General Hospital, Providence VA Medical Center, and Brown University. The participant gave informed consent to the study and the resulting publications. This data has been previously published in Rubin, et al., *J Neurosci* 2022⁶².

Human data acquisition

During recording, neural activity was recorded at 30 kHz from 192 channels and processed using a custom signal processing system. After digital downsampling, threshold crossing events were extracted in real time using 20 ms time-steps.

Human iBCI experimental paradigm

The participant first completed a center-out calibration task to optimize a Kalman filter to be used for subsequent tasks. Kalman filter weights were obtained using the rapid calibration procedure described previously⁶³. The participant then rested for a 30 min period during which baseline neural activity was recorded.

The participant then completed a number of blocks of a "Simon" memory matching task. During each trial, the researcher displayed a sequential illumination of one of four colored targets on the screen. Each illumination was 400 ms in duration and accompanied by a distinct tone. The participant was asked to imagine moving his right hand to drive the cursor from the center of the screen out to each target in the order presented. After dwelling on the correct target for 300 ms, the acquisition was deemed successful, and the cursor was automatically recentered at the origin. If the participant completed the sequence of four target acquisitions successfully, a tone indicated success, and the next trial's sequence was presented. A failed trial occurred if the cursor was dwelled on an incorrect target or if the participant failed to move the cursor to any target within 5 s.

Each trial's sequence always used all four polygons exactly once. Each block contained 16 trials, with one sequence provided per trial. In any given block, 12 of the 16 total trials were a fixed "target" sequence. The target sequence was different for each recording session but was consistent across blocks within a given session. The remaining 4 trials were a randomly chosen distractor sequence. The distractor sequences were made up of any of the other possible 23 four-target sequences. The participant was not made aware that there was a target sequence. Between each block, the participant was given the opportunity to pause or take a short break if needed.

For purposes of this study, each individual movement in the sequence of 4 movements was considered separately. Only successful movements were considered, and any movement that was part of a failed trial was discarded. We defined the intended cursor position as the straight line distance from the center of the screen to the center of

the target, which progresses linearly in time for 500 ms, and the velocity as a constant speed over this time period.

Ethics

Every experiment involving animals, human participants, or clinical samples have been carried out following a protocol approved by an ethical commission. Each human participant gave informed written consent.

Data preprocessing

We apply the following preprocessing steps to the data for all methods compared:

Resampling. In order to resample the data to larger bin sizes than originally provided (e.g., to work with 20 ms bins from 1 ms bins), we must handle both spiking data and continuous-valued data. For spiking data, we aggregate bins by summing the number of spikes. For continuous-valued data, we apply a Chebychev filter (order = 500) for anti-aliasing, and then downsample the data to the appropriate sampling rate.

Highly correlated channel removal. In order to prevent overfitting to correlated noise events across channels, we remove channels that are involved in many high correlations with other channels. We first compute the cross-correlations between all pairs of channels. We set a threshold above which we want to remove correlations. For the monkey datasets, we set a threshold of 0.2 (computed when the data is in 1 ms bins). For the T11 data, we set a threshold of 0.6 (computed when the data is in 20 ms bins). We remove any channel that is involved in a correlated pair above this threshold.

Behavioral outlier removal. For monkey datasets with continuous behavioral variables (e.g., force, cursor position), behavior was evaluated for values that fall far outside of the distribution of values for a typical trial. These outlier values can lead to errors in training the behavioral readout of the LFADS models, ADAN and Aligned FA models, and decoders. We determined cutoffs for removing values that were applied to all datasets during dataset loading. Force values are in units of voltage directly from the load cell, and cursor position values are in terms of the handle position in units of distance. These cutoffs are shown in Supplementary Table 1.

The following preprocessing step was incorporated into the NoMAD architecture only:

Normalization. To account for large changes in the firing rates of individual channels across days, we normalized each channel to have zero mean and unit standard deviation. On the continuous spiking data, we first smooth the data with a 20 ms Gaussian kernel. On the smoothed spiking data, we compute a per-channel mean and standard deviation. These means and standard deviations are saved for every day so that they can be applied to the LFADS input spiking data (not smoothed) before the low-dimensional read-in layer. Non-normalized input data is maintained so that reconstruction cost can be computed on binned spikes. We confirmed that this procedure does not unfairly bias NoMAD to perform better by presenting a sample of results without normalization (Supplementary Fig. 14).

NoMAD Architecture and parameters

The NoMAD architecture consists of two stages: fitting an initial core Day 0 LFADS model on training data from an initial day and aligning that model to data from a subsequent Day K.

Day 0 Model architecture. The LFADS model has been detailed previously^{22,31,34}. Briefly, LFADS is an instantiation of a variational autoencoder (VAE) extended to sequences. An Encoder RNN

(implemented using GRU units) takes as input a data sequence $\mathbf{X}(t)$, and produces as output a conditional distribution over a latent code Z , $Q(Z|\mathbf{X}(t))$. In the VAE framework, an uninformative prior $P(Z)$ on this latent code serves as a regularizer, and divergence from the prior is discouraged via a training penalty that scales with $\text{KL}(Q(Z|\mathbf{X}(t)) \| P(Z))$. A data sample is then drawn from $Q(Z|\mathbf{X}(t))$, which sets the initial state of a Generator RNN. This RNN attempts to create a reconstruction $\mathbf{R}(t)$ of the original data via a low-dimensional set of factors $\mathbf{F}(t)$. Specifically, the data $\mathbf{X}(t)$ are assumed to be samples from an inhomogeneous Poisson process with underlying rates $\mathbf{R}(t)$. This basic sequential autoencoder is appropriate for neural data that is well-modeled as an autonomous dynamical system. In all applications listed, we used the modified sequential autoencoder that was adapted for modeling input-driven dynamical systems. This model contains an additional Controller RNN, which compares an encoding of the observed data with the output of the Generator RNN, and attempts to inject a time-varying input $\mathbf{U}(t)$ into the Generator to account for data that cannot be modeled by the Generator's autonomous dynamics alone.

The LFADS objective function is defined as the log likelihood of the data (given the Poisson process assumption above), marginalized over all latent variables. This is optimized in the VAE setting by maximizing a variational lower bound on the marginal data log-likelihood. In training, the objective function is optimized using stochastic gradient descent, where the network parameters are updated through backpropagation through time. We used the Adam optimizer to optimize the objective function and implemented gradient clipping to prevent potential exploding gradient issues. To prevent the potentially problematic large values in RNN state variables and achieve more stable training, we also limited the range of the values in the GRU hidden state by clipping values greater than 5 and lower than -5.

In addition, a behavioral readout matrix is optionally trained to learn a mapping from the Generator RNN states to the continuous behavior. This matrix is trained to minimize the mean squared error between the true behavior and the predicted behavior. This objective function is again trained using stochastic gradient descent and backpropagation through time. A second Adam optimizer with its own learning rate and decay schedule is used to train this matrix. A separate optimizer was necessary in order for the behavioral read-out matrix training objective to be sufficiently minimized, as this training objective did not converge at the same rate as the LFADS training objective. Without its own learning rate schedule, behavioral predictions from the read-out matrix were insufficient to indicate complete learning of the mapping between the Generator states and behavior.

Day 0 Model training & hyperparameters. All experimental data is modeled without regard to trial structure, i.e., the optimization process is completely unsupervised at all stages. To do so, the continuous data (an entire session) is divided into segments. For model training, we used segments defined as follows: for monkey datasets, length 600 ms with 120 ms of overlap between segments (i.e., with our bin size of 20 ms, segments of length 30 bins with 6 bins of overlap) and for human datasets, length 1000 ms with 350 ms overlap between segments (i.e., with 20 ms bins, segments of length 50 with 17 bins of overlap). For model validation, 20% of these segments are reserved. During inference, these segment lengths are disregarded (see *Causal Inference*). We additionally enforce that no more than 5% of the validation data overlaps with the training data. We do so by selecting validation data in contiguous blocks so that validation segments overlap primarily with other validation segments.

In the Day 0 architecture, a few critical hyperparameters define the model, which we list in Supplementary Table 2.

Some datasets were fit using single LFADS models with additional behavioral readout matrices. The goal of the behavioral readout matrices was to ensure that manifolds learned on Day 0 would be highly predictive of behavior. To do this, we trained a matrix

transformation from the Generator states to the continuous-valued behavior. For the isometric task, the behavior being predicted by the readout included both force and the derivative of force; for the reaching task, the behavior included cursor position and velocity. All behavior was mean-centered and scaled to have a minimum value of -1 and maximum value of 1 prior to modeling. The T11 LFADS models were fit without a behavioral readout.

For monkey dataset experiments, single LFADS models were trained on 1 GPU each. Training stopped when there was no improvement in the performance for 10 subsequent epochs or when the learning rate reached a value of $1e-6$ during the annealing process. We found hyperparameters that were able to train models for both the isometric and kinematic datasets using grid searches. We performed a grid search over a set of values for a single hyperparameter for a subset of isometric and kinematic monkey datasets to select the best value, and repeated for all hyperparameters. We then trained all Day 0 models for monkey datasets using the hyperparameters in Supplementary Table 3.

Because it was not clear how the hyperparameters from the monkey data would translate to human data, we trained AutoLFADS models on human datasets to efficiently search the hyperparameter space. AutoLFADS is an automated hyperparameter tuning approach that trains many models (here, 20 workers) in parallel with different hyperparameters. After a set number of steps (here, 25 epochs), higher-performing models are selected to carry on. Models are also “mutated” to expand the searched hyperparameter space. This process converges to a high-performing model. While more resource-intensive, this process can be more convenient when training a smaller number of models. We detail the searched ranges in Supplementary Table 4; unless otherwise specified, other HPs can be assumed to be fixed at the values in Supplementary Table 3.

Data augmentation as regularization. In order to prevent models from overfitting to individual spikes or fast oscillations in the data, a data augmentation strategy for discrete data known as ‘spike jittering’ is applied during training. In this approach, the training procedure shifts spikes randomly in time, up to 2 bins (a settable hyperparameter) before or after their original time bin, prior to modeling the data. This augmentation is applied only to LFADS, as ADAN uses a single-time-step Day 0 model fit on Gaussian-smoothed spikes and is not prone to this type of overfitting.

Alignment model training & hyperparameters. Since we must compare data distributions from the first day to distributions from subsequent days when performing alignment, the NoMAD computational graph must contain two data flow pathways. The first pathway sends data from the first day directly through the core LFADS model. The second pathway sends data from a subsequent day through an aligner and then through the same LFADS model as the first day.

After training the LFADS model for Day 0, we trained the linear read-in matrix, the alignment network (2-layer Dense network with ReLU activations and identity initialization), the linear readout matrix from Generator states to factors, and the linear readout matrix from LFADS factors to inferred firing rates for subsequent sessions. All remaining model components had weights that were held fixed.

For the base Day 0 LFADS model (without the alignment network), and Day K model (which includes the alignment network), we separately obtained the distribution of the samples from each dimension of the Generator states for all time points, and across the entire batch of data. We then calculated the Kullback-Leibler Divergence (KL cost) between these two full-dimensional distributions, assuming they follow Multivariate Normal (Gaussian) distributions with potential correlations between each dimension. Therefore, for the KL calculation, we obtained the mean ($\mu-(0,1)$) and covariance matrices ($\Sigma-(0,1)$) of the two m -dimensional distributions (N_O , N_K) and used them to

calculate the KL divergence through Eq. (1):

$$D_{\text{KL}}(N_0||N_K) = \frac{1}{2} \left(\text{tr}(\Sigma_K^{-1}\Sigma_0) + (\mu_K - \mu_0)^\top \Sigma_K^{-1}(\mu_K - \mu_0) - m + \ln(\det \Sigma_K / \det \Sigma_0) \right) \quad (1)$$

where m is the number of Generator RNN units (here, $m=100$). Reconstruction cost is also applied to model training as described in the original LFADS paper.

We used the Adam optimizer with gradient clipping to optimize the total alignment training loss. Total loss is obtained by a weighted sum of the above KL cost and reconstruction cost. During the training, the learning rate was annealed, i.e., it was decreased through multiplication by a constant factor of 0.95 every time there was no improvement in the validation loss for a certain number of training epochs. The training procedure stops when validation loss has not shown any improvement for a fixed number of consecutive epochs. We selected the model weights corresponding to the lowest validation loss as the final model weights for inference.

The alignment model is trained by processing Day K data with the same chop lengths and overlaps as was used for Day 0. We again reserve 20% of the chops for model validation and enforce that no more than 5% of the validation data overlaps with the training data.

Hyperparameters used for NoMAD Day K training are shown in Supplementary Table 5, where M denotes hyperparameters used only for monkey datasets, H denotes hyperparameters used only for human datasets, and no denotation indicates a hyperparameter was used for all datasets.

Causal inference. Standard LFADS models, as presented in Pandarinath et al.²² perform inference by looking at an entire sequence of data to determine the initial state; hence, inference for some time points is acausal as it can be informed by time points that occur in the relative future. In order to best simulate real-time usage of our models, we perform inference in a causal manner for both Day 0 LFADS models and Day K NoMAD models. The inference approach used here uses a sliding window of observed data, where the majority of this window consists of previously observed data³⁸. At each time step, one new bin of input data is added to the window, resulting in one new bin of the model's inferred output. In addition, for computational efficiency that would be used in an online iBCI scenario, rather than sampling from the posterior distribution many times and averaging, we instead simply use the means of the posterior distributions themselves. These modifications ensure minimal latency and help us to best simulate an online real-time iBCI scenario. All figures use this procedure except for the LFADS Stitching demonstration in Supplementary Fig. 11.

We benchmarked LFADS inference latencies for a range of RNN sizes (including Encoders, Generator, and Controller RNNs) and input sequence lengths in Supplementary Fig. 13. This benchmark used 256 channels of data binned at 20 milliseconds, thereby testing the latency of LFADS when processing more channels than the 96-channel data used in this paper. This benchmark was run on a Dell Optiplex 7000 small form factor PC with an Intel i9-13900 processor and 128 GB of memory. In addition, Ali et al 2024^{J. Neural Eng.}⁶¹ benchmarked the latency of an LFADS model and found it to range from 3.5 to 5.5 ms per sample. Scaling of LFADS inference times with sequence length has also been previously documented³⁸. Inference times for single-time-step approaches (Aligned FA, ADAN) are expected to be lower than for LFADS, as they use fewer sequential operations.

Comparisons

Degenhart et al., 2020 Aligned factor analysis (FA) approach. The Degenhart et al. algorithm uses the following high-level procedure. First, the method fits a “Baseline Stabilizer” on the initial data, retaining some number of latents. This relies on factor analysis,

which is not guaranteed to converge to an optimal representation. Thus, their approach fits multiple FA models with random initialization, and they select the model with the highest log-likelihood. After fitting the baseline stabilizer, they also fit the data to be aligned with an FA model (same procedure as step 1). Next, they identify stable loading rows between the two models. This consists of iteratively trying to align the two loading matrices. After each alignment, they identify rows that are the most different after each alignment, and remove them. Finally, they learn the optimal orthonormal transformation to align the identified stable rows (i.e., solving the “Procrustes problem”). For our comparisons, we used the parameters in Supplementary Table 6.

Based on data presented in Ma*, Rizzoglio*, et al.¹⁹, smoothing binned spike data prior to alignment with this approach improves performance¹⁹. Therefore, we use 20 ms binned spike data smoothed with a 40 ms Gaussian kernel as input data to this method. Only data from within behavioral trials is used to train this method, and behavioral trials containing outliers, as determined in *Behavioral Outlier Removal*, were discarded. We train the baseline stabilizer on 80% of the trial data from Day 0 and the aligned stabilizer on 80% of the trials from Day K. We evaluate decoding on the resultant latent factors as described in *Neural Decoding*.

Adversarial domain adaptation network (ADAN). This method begins by fitting an autoencoder to reproduce smoothed binned spiking data and an RNN decoder to predict force (isometric monkey) or cursor velocity (kinematic monkey) activity from the manifold. To ensure a good Day 0 fit, we train this autoencoder using 5-fold cross-validation and select the model with the best behavioral R^2 . Then, ADAN is trained in a method similar to that of generative adversarial networks (GANs). A discriminator network is an autoencoder that acts to maximize the difference between the neural reconstruction losses on the two days. The distribution alignment module (the generator) works against the discriminator by minimizing the neural reconstruction losses on Day K. This results in alignment of the Day K manifold to the Day 0 manifold. We train ADAN consistently with the Ma*, Rizzoglio*, et al. paper¹⁹, in which ADAN training is paired with behavioral predictions; however, we report decoding accuracy from a separate Wiener filter in order to use a consistent decoding approach across all compared methods (see *Neural Decoding*). We use the parameters detailed in Supplementary Table 7 to train ADAN.

As input data to this method, we use 20 ms binned spiking data smoothed with a 40 ms Gaussian kernel. Only data from within behavioral trials is used to train this method, and behavioral trials with outliers are discarded (see *Behavioral Outlier Removal*).

Static decoder. Binned spikes (20 ms bins) were smoothed with a Gaussian kernel (40 ms). A decoder was trained on the Day 0 smoothed spikes. This fixed decoder was applied to the Day K smoothed spikes and evaluated.

Supplementary comparisons

Targeted neural dynamical modeling (TNDM). First, we train a TNDM model on Day 0. Then, we use this trained model with fixed weights to perform inference and obtain weight predictions on a Day K dataset. We repeat this for all pairs of datasets for both monkeys and report decoding accuracy using a Wiener filter trained on the resulting Day 0 inferred firing rates. We sourced our TNDM implementation from the following repository: <https://github.com/HennigLab/tndm>. Because this implementation only supports the modeling of autonomous dynamical systems (e.g., it lacks a Controller RNN to model inferred inputs), we used within-trial data aligned to the movement onset (see *Neural Decoding*) as input data. Behavioral data was provided to the Day 0 model in the same form as was provided to Day 0 LFADS models. As no early stopping metrics are implemented in this repository, we

use the provided default of 1000 training epochs for each model. We use the hyperparameters in Supplementary Table 8 to train TNDM.

Adaptive LFADS. On Day 0, we train an LFADS model with lowD read-in but without the behavioral readout matrix. On Day K, we again hold the weights of the core LFADS model fixed and allow only the read-in and read-out matrices to train. No alignment network or KL Divergence cost are included. Hyperparameters for both Day 0 and Day K model training are the same as those laid out in Supplementary Tables 2–3. We repeat this training procedure for all pairs of datasets and report decoding results using Wiener filter decoders trained on the Day 0 LFADS factors.

Retrospective target inference (RTI). This analysis aimed to demonstrate that RTI and manifold alignment approaches are complementary when applied within a single session. As a result, we identified iBCI data during which a participant is performing closed-loop control in which the intended action can be inferred post-hoc (e.g., selecting among a defined set of targets). In addition, we looked for datasets during which the participant is performing the same behavior over a long recording period (many minutes to hours), to allow sufficient data for both NoMAD and RTI to be performed at multiple timepoints throughout the session. During this long recording period, it was also important that no decoder recalibration occurred, so that there were no changes that would alter the relationship between neural activity and the participant's control strategy. We found two sessions that met these criteria that we were able to use for our RTI analysis. Our RTI procedure was as follows: On Block 0, we fit a decoder using the neural data and estimated intended behavior. We apply this decoder as-is to the next available block. We then retrain the decoder to include the neural data and estimated intended behavior from both previous blocks. We apply this scheme moving forward, applying the decoder to Block K that has been trained using Blocks 0 through K-1.

Neural decoding

The original Aligned FA demonstration applied a linear decoder (steady-state Kalman filter) to the aligned latents⁶⁴. Similarly, while the original ADAN work used an LSTM decoder¹⁸, more recent applications of ADAN from the same team use a linear decoder (Wiener filter)¹⁹. In our own testing, we found that using an LSTM with ADAN did not offer any performance benefits over a Wiener filter, and also made training ADAN more difficult. Thus, after these tests and consultation with the authors of ADAN, we focused on linear decoding, as described below.

Wiener filter. For both the kinematic and isometric datasets, prediction of behavioral output was done using a Wiener filter. Wiener filters predict the current value of an output signal using previous timesteps, as defined by Eq. (2):

$$\mathbf{y}[t] = \sum_{i=0}^{I-1} w_i \mathbf{x}[t-i] \quad (2)$$

where $\mathbf{y}[t]$ is the output signal at time t , $\mathbf{x}[t]$ is the input signal at time t , w_i is the filter coefficient, and I is the number of previous samples to use for decoding. In our decoder, the input signal \mathbf{x} is the (aligned) manifold, \mathbf{y} is the behavioral output to predict, and I is the number of time bins of history (set to 4 for monkey datasets and 1 for the T11 dataset). We present a comparison of decoders with $I=4$ and $I=1$ (i.e., an optimal linear estimator (OLE)) when decoding the primate datasets in Supplementary Fig. 15. The weights are fit using a matrix formation of the above equation as defined in Eq. (3):

$$\mathbf{W} = (\mathbf{X}^T \mathbf{X} + \mathbf{R}^T \mathbf{R})^{-1} \mathbf{X}^T \mathbf{y} \quad (3)$$

where \mathbf{W} is a matrix of filter coefficients, \mathbf{X} represents the predictor data with history and bias, and \mathbf{y} represents the output signal. \mathbf{R}

represents a diagonal matrix with the L2 regularization constant filling the diagonal. The bias term is not regularized and therefore its diagonal entry is set to zero.

The L2 regularization aims to avoid decoder overfitting by penalizing solutions with large individual weights. L2 regularization values are obtained using 10-fold cross-validation. We sweep a range of 20 values spanning $1\text{e}1$ to $1\text{e}5$ in logspace. For each value, we train and test a Wiener filter using 10-fold cross-validation, testing the decoder on a held-out fold. The optimal regularization value was selected based on which value yielded the highest performance metric. After training on the Day 0 dataset, the final performance was reported on the held-out fold. No Day K data is used for decoder training; therefore, the choice of held-in and held-out data used for Day 0 training do not affect performance reported on Day K.

As input to the Wiener Filter, we use the latent states (\mathbf{x}) and behavior (\mathbf{y}). The dimensionalities of the latent states are as follows: 100 for NoMAD, 10 for ADAN, and 10 for Aligned FA (see Supplementary Fig. 16 for justification).

For monkey datasets, each trial was represented as a window 250 ms before to 500 ms after movement onset. For each trial, the movement onset point was calculated using the period 250 ms before the go cue to 750 ms after the go cue. We first searched within this time period to identify the point at which the cursor reaches its maximum speed. From that point, we searched backwards in time to identify the point at which the cursor last reached 20% of its maximum speed. In parallel, we searched forward in time, beginning at the go cue, to find the point at which the cursor first reached 20% of its maximum speed. These two points should be consistent—if not, it indicates a trial in which the monkey started a movement, stopped, and then started again. We rejected trials with inconsistent movement onset calculations. We further rejected trials in which the backward move onset (last time the cursor reaches 20% of max speed) occurred before the point at which the target is displayed—this typically indicated that the monkey had not yet begun its movement in the time period analyzed, potentially due to inattention. While these trial rejections have minor effects on the analysis, we performed them to ensure the decoding metrics were a consistent and robust indicator of each method's performance. Only trials for which the monkey successfully completed the trial and movement onset was successfully calculated were considered. To account for drifts in the behavioral data baseline prior to movement onset that occurred across days, we manually set the first point in each trial's force or velocity to zero and shift the remaining points accordingly.

For the T11 dataset, a trial is defined as the region from the start of one of the four movements in a trial to 600 ms after that start point or to the point where the target is acquired and the cursor is automatically returned to center (whichever happens first). The timing of each movement is isolated as follows. First, we detect the automatic return to center that occurs when a target is acquired by the participant. We do this by considering the gradient of the position and detecting the nonlinearities that indicate the jump in position from the target location to the center. These are then considered the stop times of each movement. From there, we look backwards in time for the nearest location where the position crosses the center point. We consider the time of this zero-crossing to be the start time of the movement. We only consider movements that are part of successful trials—that is, all four targets in a trial need to be successfully acquired for us to include the reach in our analysis.

Evaluation of decoding performance. The accuracy of neural decoding was measured using R^2 , defined as in Eq. (4):

$$R^2(\mathbf{y}, \hat{\mathbf{y}}) = 1 - \frac{\sum_{d=1}^D \sum_{i=1}^N (\hat{y}_{i,d} - y_{i,d})^2}{\sum_{d=1}^D \sum_{i=1}^N (y_{i,d} - \bar{y}_d)^2} \quad (4)$$

where D is the number of dimensions of the predicted output, N is the number of data samples, $y_{i,d}$ is an actual data sample for one dimension, \bar{y}_d is the mean of the actual signal in one dimension, and $\hat{y}_{i,d}$ is a predicted data sample for one dimension. In practice, we used the function `sklearn.metrics.r2_score(y, y_hat, multioutput='variance_weighted')`⁶⁵.

Manifold visualizations

Dimensionality reduction. To create visualizations of the manifold at different stages of the alignment process, we used demixed principal components analysis (dPCA)³⁹. We applied regularized dPCA on the Day 0 manifold. We restricted dPCA fitting to successful trials within the window 250 ms before to 500 ms after target onset. After learning the Day 0 dPCA transformation, we applied the same transformation to the Day K manifold using both the Day 0 dPCA weights and the Day 0 mean offsets.

Visualizations were created by plotting the top condition-independent components and the top two condition-dependent components, as ranked by variance explained. This allows for the comparison of Day 0 to Day K before and after alignment without dependence on neural decoding or behavior.

Alignment quantification. To assess the extent of alignment between the Day K manifolds at each stage and the Day 0 manifold, we computed the Euclidean distance between the condition-averaged dPCs in three dimensions.

Calculating decline in decoding performance over months

To quantify how much the decoding performance declined over the available timespan, we computed the rate of decay of the median performance within each 5-day bin. In order to reduce the effect of highly negative data points, we first convert R^2 to a signal-to-noise ratio (SNR) metric using Eq. (5):

$$\text{SNR} := -10\log_{10}(1 - R^2) \quad (5)$$

as described in Makin et al.⁶⁶. We then fit an exponential decay equation of the form $y = Ae^{-Bt}$ to the data where y is the median SNR within each 5-day bin with the median value of the Day 0 within-day performance appended as the first data point, and t is the middle of each bin with $t=0$ appended as the first data point. This fits the constants A and B , where B is the decay constant in units of 1/days. We convert this to half-life in days for ease of interpretation using the formula $1/B * \ln(2)$.

Computing significance

Significant differences between decoding results were quantified using Wilcoxon signed-rank tests as defined by the function `scipy.stats.wilcoxon`.

Reporting summary

Further information on research design is available in the Nature Portfolio Reporting Summary linked to this article.

Data availability

The monkey data analyzed in this manuscript were made available in a previous publication^{19,67}. The monkey isometric force task data (Jango_ISO_2015) and unloaded reaching task data (Chewie_CO_2016) used in this study are available in the Dryad database (<https://doi.org/10.5061/dryad.cvdncjt7n>). The human BCI participant data used in this study are also available in the Dryad database (<https://doi.org/10.5061/dryad.q83bk3jtp>). All data supporting the findings of this study are available within the article and its supplementary files. Any additional requests for information can be directed to and will be

fulfilled by the corresponding authors. Source data are provided in this paper.

Code availability

Code for the method developed in this study is available at <https://github.com/snel-repo/nomad> with license details described.

References

- Collinger, J. L., Gaunt, R. A. & Schwartz, A. B. Progress towards restoring upper limb movement and sensation through intracortical brain-computer interfaces. *Curr. Opin. Biomed. Eng.* **8**, 84–92 (2018).
- Pandarinath, C. & Bensmaia, S. J. The science and engineering behind sensitized brain-controlled bionic hands. *Physiol. Rev.* **102**, 551–604 (2022).
- Ajiboye, A. B. et al. Restoration of reaching and grasping movements through brain-controlled muscle stimulation in a person with tetraplegia: a proof-of-concept demonstration. *Lancet* **389**, 1821–1830 (2017).
- Collinger, J. L. et al. High-performance neuroprosthetic control by an individual with tetraplegia. *Lancet* **381**, 557–564 (2013).
- Wodlinger, B. et al. Ten-dimensional anthropomorphic arm control in a human brain-machine interface: difficulties, solutions, and limitations. *J. Neural Eng.* **12**, 016011 (2014).
- Willett, F. R., Avansino, D. T., Hochberg, L. R., Henderson, J. M. & Shenoy, K. V. High-performance brain-to-text communication via handwriting. *Nature* **593**, 249–254 (2021).
- Perge, J. A. et al. Intra-day signal instabilities affect decoding performance in an intracortical neural interface system. *J. Neural Eng.* **10**, 036004 (2013).
- Downey, J. E., Schwed, N., Chase, S. M., Schwartz, A. B. & Collinger, J. L. Intracortical recording stability in human brain-computer interface users. *J. Neural Eng.* **15**, 046016 (2018).
- Wimalasena, L. N., Miller, L. E. & Pandarinath, C. From unstable input to robust output. *Nat. Biomed. Eng.* **4**, 665–667 (2020).
- Sussillo, D., Stavisky, S. D., Kao, J. C., Ryu, S. I. & Shenoy, K. V. Making brain-machine interfaces robust to future neural variability. *Nat. Commun.* **7**, 13749 (2016).
- Huggins, J. E., Moinuddin, A. A., Chiodo, A. E. & Wren, P. A. What Would Brain-Computer Interface Users Want: Opinions and Priorities of Potential Users With Spinal Cord Injury. *Arch. Phys. Med. Rehabil.* **96**, S38–S45.e5 (2015).
- Silversmith, D. B. et al. Plug-and-play control of a brain-computer interface through neural map stabilization. *Nat. Biotechnol.* **39**, 326–335 (2020).
- Churchland, M. M. et al. Neural population dynamics during reaching. *Nature* **487**, 51–56 (2012).
- Yu, B. M. et al. Gaussian-process factor analysis for low-dimensional single-trial analysis of neural population activity. In *Advances in Neural Information Processing Systems* **21** (Curran Associates, Inc., 2008).
- Cunningham, J. P. & Yu, B. M. Dimensionality reduction for large-scale neural recordings. *Nat. Neurosci.* **17**, 1500–1509 (2014).
- Dabagia, M., Kording, K. P. & Dyer, E. L. Comparing high-dimensional neural recordings by aligning their low-dimensional latent representations. Preprint at <https://doi.org/10.48550/arXiv.2205.08413> (2022).
- Degenhart, A. D. et al. Stabilization of a brain-computer interface via the alignment of low-dimensional spaces of neural activity. *Nat. Biomed. Eng.* **4**, 672–685 (2020).
- Farshchian, A. et al. Adversarial Domain Adaptation for Stable Brain-Machine Interfaces. In *Proceedings of The Seventh International Conference on Learning Representations (ICLR)* (2019).
- Ma, X. et al. Using adversarial networks to extend brain computer interface decoding accuracy over time. *eLife* **12**, e84296 (2023).

20. Bishop, W. et al. Self-recalibrating classifiers for intracortical brain-computer interfaces. *J. Neural Eng.* **11**, 026001 (2014).
21. Gallego, J. A., Perich, M. G., Chowdhury, R. H., Solla, S. A. & Miller, L. E. Long-term stability of cortical population dynamics underlying consistent behavior. *Nat. Neurosci.* **23**, 260–270 (2020).
22. Pandarinath, C. et al. Inferring single-trial neural population dynamics using sequential auto-encoders. *Nat. Methods* **15**, 805–815 (2018).
23. Kao, J. C., Ryu, S. I. & Shenoy, K. V. Leveraging neural dynamics to extend functional lifetime of brain-machine interfaces. *Sci. Rep.* **7**, 7395 (2017).
24. Pandarinath, C. et al. Latent factors and dynamics in motor cortex and their application to brain-machine interfaces. *J. Neurosci.* **38**, 9390–9401 (2018).
25. Kao, J. C. et al. Single-trial dynamics of motor cortex and their applications to brain-machine interfaces. *Nat. Commun.* **6**, 7759 (2015).
26. Keshtkaran, M. R. et al. A large-scale neural network training framework for generalized estimation of single-trial population dynamics. *Nat. Methods* **19**, 1572–1577 (2022).
27. Zhu, F. et al. Deep inference of latent dynamics with spatio-temporal super-resolution using selective backpropagation through time. In *Proceedings of Advances in Neural Information Processing Systems*. **34** (2021).
28. Pei, F. et al. Neural Latents Benchmark '21: Evaluating latent variable models of neural population activity. In *Proceedings of Advances in Neural Information Processing Systems*. **35** (2022).
29. Dyer, E. L. et al. A cryptography-based approach for movement decoding. *Nat. Biomed. Eng.* **1**, 967–976 (2017).
30. Lee, J., Dabagia, M., Dyer, E. & Rozell, C. Hierarchical optimal transport for multimodal distribution alignment. In *Proceedings Advances in Neural Information Processing Systems*. **32** (2019).
31. Sussillo, D., Jozefowicz, R., Abbott, L. F. & Pandarinath, C. LFADS - Latent Factor Analysis via Dynamical Systems. Preprint at <https://doi.org/10.48550/arXiv.1608.06315> (2016).
32. Chung, J. et al. A recurrent latent variable model for Sequential Data. In *Proceedings Advances in Neural Information Processing Systems* **28** (2016).
33. Gregor, K., Danihelka, I., Graves, A., Rezende, D. J. & Wierstra, D. DRAW: A Recurrent Neural Network For Image Generation. *Proceedings of the 32nd International Conference on Machine Learning (PMLR)*. **37**, 1462–1471 (2015).
34. Keshtkaran, M. R. & Pandarinath, C. Enabling hyperparameter optimization in sequential autoencoders for spiking neural data. In *Proceedings Advances in Neural Information Processing Systems* **2** (2019).
35. Sani, O. G., Abbaspourazad, H., Wong, Y. T., Pesaran, B. & Shanechi, M. M. Modeling behaviorally relevant neural dynamics enabled by preferential subspace identification. *Nat. Neurosci.* **24**, 140–149 (2021).
36. Hurwitz, C. et al. Targeted neural dynamical modeling. In *Proceedings of Advances in Neural Information Processing Systems*. **35**, (2022).
37. Sani, O. G., Pesaran, B. & Shanechi, M. M. Where is all the non-linearity: flexible nonlinear modeling of behaviorally relevant neural dynamics using recurrent neural networks. Preprint at <https://doi.org/10.1101/2021.09.03.458628> (2021).
38. Ye, J. & Pandarinath, C. Representation learning for neural population activity with Neural Data Transformers. *Neurons, Behavior, Data analysis, and Theory*. **5**, 3 (2021).
39. Kobak, D. et al. Demixed principal component analysis of neural population data. *eLife* **5**, e10989 (2016).
40. Perich, M. G., Gallego, J. A. & Miller, L. E. A neural population mechanism for rapid learning. *Neuron* **100**, 964–976 (2018).
41. Jarosiewicz, B. et al. Virtual typing by people with tetraplegia using a self-calibrating intracortical brain-computer interface. *Sci. Transl. Med.* **7**, 313ra179 (2015).
42. Wilson, G. H. et al. Long-term unsupervised recalibration of cursor BCIs. Preprint at <https://doi.org/10.1101/2023.02.03.527022> (2023).
43. Li, Z., O'Doherty, J. E., Lebedev, M. A. & Nicolelis, M. A. L. Adaptive decoding for brain-machine interfaces through bayesian parameter updates. *Neural Comput.* **23**, 3162–3204 (2011).
44. Cunningham, J. P. et al. A closed-loop human simulator for investigating the role of feedback control in brain-machine interfaces. *J. Neurophysiol.* **105**, 1932–1949 (2011).
45. Bengio, Y., Courville, A. & Vincent, P. Representation learning: A review and new perspectives. *IEEE Trans. Pattern Anal. Mach. Intell.* **35**, 1798–1828 (2014).
46. Deo, D. R. et al. Brain control of bimanual movement enabled by recurrent neural networks. *Sci. Rep.* **14**, 1598 (2024).
47. Flint, R. D. et al. The representation of finger movement and force in human motor and premotor cortices. *eNeuro* **7**, <https://doi.org/10.1523/eneuro.0063-20.2020> (2020).
48. Flint, R. D., Wright, Z. A., Scheid, M. R. & Slutzky, M. W. Long term, stable brain machine interface performance using local field potentials and multiunit spikes. *J. Neural Eng.* **10**, 056005 (2013).
49. Schwartz, A. B., Cui, X. T., Weber, D. J. & Moran, D. W. Brain-controlled interfaces: Movement restoration with neural prosthetics. *Neuron* **52**, 205–220 (2006).
50. Chung, J. E. et al. High-density, long-lasting, and multi-region electrophysiological recordings using polymer electrode arrays. *Neuron* **101**, 21–31 (2019).
51. Rousche, P. J. et al. Flexible polyimide-based intracortical electrode arrays with bioactive capability. *IEEE Trans. Biomed. Eng.* **48**, 361–371 (2001).
52. Sohal, H. S. et al. The sinusoidal probe: a new approach to improve electrode longevity. *Front. Neuroeng.* **7**, 10 (2014).
53. Luan, L. et al. Ultraflexible nanoelectronic probes form reliable, glial scar-free neural integration. *Sci. Adv.* **3**, e1601966 (2017).
54. Wei, X. et al. Nanofabricated ultraflexible electrode arrays for high-density intracortical recording. *Adv. Sci.* **5**, 1700625 (2018).
55. Sadtler, P. T. et al. Neural constraints on learning. *Nature* **512**, 423–426 (2014).
56. Oby, E. R. et al. New neural activity patterns emerge with long-term learning. *Proc. Natl. Acad. Sci. USA* **116**, 15210–15215 (2019).
57. Ma, X., Bodkin, K. L. & Miller, L. E. Population Activity in Motor Cortex is Influenced by the Contexts of the Motor Behavior. In *2021 10th International IEEE/EMBS Conference on Neural Engineering (NER)* 1152–1155 (2021).
58. Chase, S. M., Schwartz, A. B. & Kass, R. E. Bias, optimal linear estimation, and the differences between open-loop simulation and closed-loop performance of spiking-based brain-computer interface algorithms. *Neural Netw.* **22**, 1203–1213 (2009).
59. Koyama, S. et al. Comparison of brain-computer interface decoding algorithms in open-loop and closed-loop control. *J. Comput. Neurosci.* **29**, 73–87 (2010).
60. Willett, F. R., Suminski, A. J., Fagg, A. H. & Hatsopoulos, N. G. Improving brain-machine interface performance by decoding intended future movements. *J. Neural Eng.* **10**, 026011 (2013).
61. Ali, Y. H. et al. BRAND: A platform for closed-loop experiments with deep network models. *J. Neural Eng.* **21**, <https://doi.org/10.1088/1741-2552/ad3b3a> (2024).
62. Rubin, D. B. et al. Learned motor patterns are replayed in human motor cortex during sleep. *J. Neurosci.* **42**, 5007–5020 (2022).
63. Brandman, D. M. et al. Rapid calibration of an intracortical brain-computer interface for people with tetraplegia. *J. Neural Eng.* **15**, 026007 (2018).

64. Malik, W. Q., Truccolo, W., Brown, E. N. & Hochberg, L. R. Efficient decoding with steady-state Kalman filter in neural interface systems. *IEEE Trans. Neural Syst. Rehabil. Eng.* **19**, 25–34 (2011).
65. Pedregosa, F. et al. Scikit-learn: Machine learning in Python. *J. Mach. Learn. Res.* **12**, 2825–2830 (2018).
66. Makin, J. G., O'Doherty, J. E., Cardoso, M. M. B. & Sabes, P. N. Superior arm-movement decoding from cortex with a new, unsupervised-learning algorithm. *J. Neural Eng.* **15**, 026010 (2018).
67. Ma, X. et al. Data from: Using adversarial networks to extend brain computer interface decoding accuracy over time. **12**, <https://doi.org/10.7554/elife.84296> (2023).
68. Ethier, C., Oby, E. R., Bauman, M. J. & Miller, L. E. Restoration of grasp following paralysis through brain-controlled stimulation of muscles. *Nature* **485**, 368–371 (2012).

Acknowledgements

The authors would like to thank Participant T11, his family and care partners, Beth Travers, Dave Rosler, and Maryam Masood for their contributions to this research. We thank Matthew Perich and Stephanie Naufel for the collection of the data used in this work. We thank Ali Farshchian for helpful discussions regarding ADAN and Fabio Rizzoglio for helpful discussions regarding ADAN and Aligned FA. This work was supported by the Emory Neuromodulation and Technology Innovation Center (ENTICE), NSF NCS 1835364, DARPA PA-18-02-04-INI-FP-021, NIH Eunice Kennedy Shriver NICHD K12HD073945, NIH-NINDS/OD DP2NS127291, NIH BRAIN/NIDA RF1 DA055667, the Alfred P. Sloan Foundation, the Burroughs Wellcome Fund, and the Simons Foundation as part of the Simons-Emory International Consortium on Motor Control (CP), NIH NINDS NS053603 and NS074044 (LEM), NIH NIBIB T32EB025816 (BMK, YHA), NSF Graduate Research Fellowship DGE-2039655 (ARS), NIH NIDCD DC017844, and Office of Research and Development, Department of Veterans Affairs A2295R and N2864C (LRH).

Author contributions

The ideas in this manuscript were conceptualized by L.E.M. and C.P. Funding was acquired by L.R.H., L.E.M., and C.P. Investigation was performed by B.M.K., Y.H.A., L.N.W., A.R.S., M.R.K., and C.P. Data was prepared by K.B., X.M., and D.B.R. The manuscript was written by B.M.K. and C.P. The manuscript was revised by B.M.K., Y.H.A., L.N.W., A.R.S., L.E.M., and C.P. The work was supervised by Z.M.W., S.S.C., L.R.H., L.E.M., and C.P.

Competing interests

The MGH Translational Research Center has a clinical research support agreement (CRSA) with Axoft, Neuralink, Neurobionics, Paradromics, Precision Neuro, Synchron, and Reach Neuro, for which LRH provides consultative input. LRH is a non-compensated member of the Board of Directors of a nonprofit assistive communication device technology foundation (Speak Your Mind Foundation). Mass General Brigham (MGB) is convening the Implantable Brain-Computer Interface Collaborative Community (iBCI-CC); charitable gift agreements to MGB, including

those received to date from Paradromics, Synchron, Precision Neuro, Neuralink, and Blackrock Neurotech, support the iBCI-CC, for which LRH provides effort. The MGH Translational Research Center has a clinical research support agreement (CRSA) with Paradromics and Ability Neurotech for which DBR provides consultative input. CP is a research scientist at Meta (Reality Labs). These entities did not support this work, did not have a role in the study, and do not have any financial interests related to this work. The algorithm presented in this work is covered by the submitted U.S. patent application no. 17/512.339, with authors C.P., M.R.K., Y.H.A., L.N.W., A.R.S., L.E.M., X.M., and B.M.K. listed as inventors. The remaining authors declare no competing interests. The content is solely the responsibility of the authors and does not necessarily represent the official views of the National Institutes of Health, or the Department of Veterans Affairs, or the United States Government. CAUTION: Investigational Device. Limited by Federal Law to Investigational Use.

Additional information

Supplementary information The online version contains supplementary material available at <https://doi.org/10.1038/s41467-025-59652-y>.

Correspondence and requests for materials should be addressed to Chethan Pandarinath.

Peer review information *Nature Communications* thanks the anonymous reviewer(s) for their contribution to the peer review of this work. A peer review file is available.

Reprints and permissions information is available at <http://www.nature.com/reprints>

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Open Access This article is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 International License, which permits any non-commercial use, sharing, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if you modified the licensed material. You do not have permission under this licence to share adapted material derived from this article or parts of it. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by-nc-nd/4.0/>.

© The Author(s) 2025