Assistive sensory-motor perturbations influence learned neural representations

Pavithra Rajeswaran[1], Alexandre Payeur[2,3], Guillaume Lajoie[2,3], and Amy L. Orsborn[1,4,5,*]

[1]University of Washington, Bioengineering, Seattle, 98115, USA
[2]Université de Montréal, Department of Mathematics and Statistics, Montréal (QC), Canada, H3C 3J7
[3]Mila - Québec Artificial Intelligence Institute, Montréal (QC), Canada, H2S 3H1
[4]University of Washington, Electrical and Computer Engineering, Seattle, 98115, USA
[5]Washington National Primate Research Center, Seattle, Washington, 98115, USA
[*]aorsborn@uw.edu

ABSTRACT

Task errors are used to learn and refine motor skills. We investigated how task assistance influences learned neural representations using Brain-Computer Interfaces (BCIs), which map neural activity into movement via a decoder. We analyzed motor cortex activity as monkeys practiced BCI with a decoder that adapted to improve or maintain performance over days. Over time, task-relevant information became concentrated in fewer neurons, unlike with fixed decoders. At the population level, task information also became largely confined to a few neural modes that accounted for an unexpectedly small fraction of the population variance. A neural network model suggests the adaptive decoders directly contribute to forming these more compact neural representations. Our findings show that assistive decoders manipulate error information used for long-term learning computations like credit assignment, which informs our understanding of motor learning and has implications for designing real-world BCIs.

We learn new motor skills by incrementally minimizing movement errors [1]. This process involves both widely distributed computations [2] and targeted neurophysiological changes [3, 4, 5]. Such coordinated circuit changes might be mediated by a poorly understood "credit assignment" process where movement errors lead to adjustment in selected neural structures [6].

Intracortical brain-computer interfaces (BCIs) provide a powerful framework to study how motor errors shape neural changes [7, 8]. BCIs define a mapping ("decoder"; Fig. 1A) between neural activity and movement [9, 10, 11] to causally interrogate learning. Perturbations of well-learned BCI mappings, analogous to perturbations of natural movement [12, 13], have been used to probe the neural mechanisms of short-term adaptation by introducing task errors [14, 15]. Alternately, novel BCI mappings learned over multiple practice sessions can be used to study neural mechanisms related to skill formation [16, 17, 18, 19]. BCI skill learning studies often use decoders that provide limited performance, requiring the brain to reduce task errors through changes in neural activity. In contrast, application-driven BCIs often manipulate decoders to *improve* or *maintain* performance over time (Fig. 1A,B) [11, 20, 21, 22, 23, 24, 25, 26, 27, 28]. These assistive interventions perturb the mapping between neural activity and movement without introducing task errors and potentially reducing them, thus providing an opportunity to study how errors influence skill learning mechanisms.

BCI skill learning alters both single-neuron and population activity. Practice with the same decoder over several days refines performance, stabilizes neural representations [16] and improves coordination among neurons [29, 30], which is consistent with

reduced neuronal variability in natural motor learning [31]. Some changes are targeted to specific neurons: neurons used as direct "readouts" of behavior are modulated differently than "nonreadout" neurons, which may participate in computations but do not drive movement directly (Fig. 1B). Preferential modulation of readouts with extended practice is consistent with the possibility that credit assignment computations contribute to skill learning [30, 32, 33, 34, 35, 36]. Population-level changes also appear to be primarily subserved by readouts [30].

Long-term learning may also occur during extended practice with assistive BCI decoders. Assistive decoders adapt the neural-movement mapping to reduce errors, and also help compensate for unstable measurements by modifying which neural units are part of the readout ensemble [28] (Fig. 1B). Despite this dynamic mapping, adaptive decoders can help produce skillful control of BCI over time[11, 21, 28, 36], and reduce the amount of neural changes needed, such as shifts in neuronal tuning [28]. Yet, longitudinal adaptive BCI experiments consistently report performance improvements beyond what would be expected from decoder adaptation alone [11, 21, 28, 36, 37, 38, 39], consistent with long-term skill learning processes taking place.

It is unclear how decoder interventions that manipulate errors impact learning and the resulting neural representations. We hypothesized that adaptive BCIs might alter feedback that contributes to the targeted neural changes that occur during skill learning. Better understanding this process could inform BCI applications that widely use these manipulations and provide insights into the computations guiding long-term sensorimotor learning. However, examining brain-algorithm interactions is challenging because it requires comparing learning outcomes between different paradigms (fixed vs. adaptive decoders) where variations in task performance, individuals, and training history complicate interpretations. Existing work has therefore focused on performance-based comparisons of different BCI algorithms without examining their impact on neural representations.

We used experimental data from longitudinal BCI learning studies [16, 28] and recurrent neural network (RNN) models to explore how adaptive decoders influence learned neural representations. The experimental data provided insights into real-world neural adaptation, while the RNN model allowed for direct comparisons that are not possible experimentally. Our findings show that learning-mediated changes were targeted to the readouts, consistent with credit assignment computations. But surprisingly, these changes were mostly concentrated in a subset of readouts when decoders adapted. In contrast, fixed-decoder experiments did not show such *compaction* with learning. Our model helped ascertain the direct involvement of decoder adaptation in the compaction process, and offered insight into how adaptive decoders could influence plasticity. Modeling also suggested opportunities to tune brain-decoder interactions to balance performance goals. Finally, data analysis of population activity revealed that learning mainly targeted low-variance components, which are often overlooked by standard methods [40]. Together, our findings provide evidence that adaptive decoders influence neural encoding on multiple levels, shedding new light on how error manipulations shape neural activity during learning.

Results

Two rhesus macaques (monkeys J and S) learned to perform point-to-point movements with a BCI controlled cursor (Fig. 1A), as previously described [28]. Multi-unit action potentials from microelectrode arrays implanted in the arm area of the primary- and premotor cortices were used as inputs to a Kalman filter decoder that controlled cursor velocity and position. A subset of all recorded units was selected as input to the decoder based on recording properties (e.g., stability over time) and defined the *readout* population (Fig. 1A) (see Methods).

Each monkey learned to control the cursor over several consecutive days, defined as a learning series. We focused on learning series lasting at least four days ($n = 7$ series for monkey J; $n = 3$ for monkey S). Learning series varied in length, initial decoder training methods, and number of decoder updates. Decoder weights were updated via Closed-Loop Decoder Adaptation (CLDA) by realigning cursor velocities produced by readouts toward the target (see Methods and Fig. S1A). CLDA was

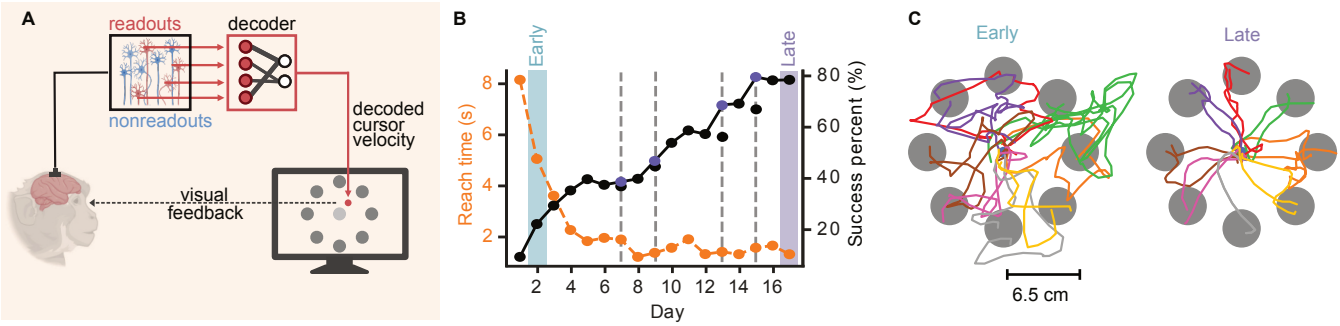

Figure 1. Experimental setup and example behavior (A) Closed-loop BCI setup. Multi-unit activities from readouts (red) and nonreadouts (blue) were recorded in a monkey's motor cortex. Only the readout activity was decoded to control a cursor in a 2D center-out reaching task with visual feedback. (Monkey and neuron icons from BioRender.) **(B)** Mean reach time (dashed orange line) and success percentage (solid black line) across days for an example learning series (monkey J). Early (blue) and late (purple) training phases are indicated. Vertical grey dashed lines indicate decoder adaptation, with purple dots showing performance gains from decoder adaptation. **(C)** Early (left) and late (right) cursor trajectories for the series in panel B.

done at the beginning of a learning series, improving initial performance to provide sufficient cursor control throughout the workspace. CLDA was also done intermittently within series to maintain performance when neural measurements varied. Longer series generally had more decoder adaptation events (Fig. S1D). As previously shown [28], task success rates and reach times improved over days (Fig. 1B, a selected series for monkey J used below in all single-series examples, for consistency; see additional example series in Fig. S1B (monkey S) and Fig. S1E (monkey J)).

We quantified learning within each series by defining an 'early' and 'late' training day, which corresponded to the first and last day of a learning series with at least 25 successful reaches per target direction, respectively (day 2 and 17 in Fig. 1B). We were primarily interested in characterizing neural representations related to movement direction, and therefore included any trial wherein the target was reached regardless of whether the hold at the target was successful. Improvements in task success rate and reach time were accompanied by more direct and accurate reaches (Fig. 1C , Fig. S1C and [28]). The dataset included series where the amount of CLDA performed on day 1 varied, resulting in varying initial performance, but behavioral improvements consistent with long-term skill learning were observed in all series [28].

Credit assignment to readout units during long-term learning with adaptive decoders

We first assessed whether adaptive decoders impacted credit assignment in motor cortex by investigating how task-relevant information was distributed between readouts and nonreadouts. Because cursor kinematics changed significantly across days, we instead used an offline classification analysis to quantify how the fixed target identities were encoded. Target identity does not change over days, facilitating cross-day comparisons despite behavioral changes (Fig. 1C). Specifically, each day we fitted a multiclass logistic regression to predict target identity using neural activity aligned on the go-cue (Fig. 2A), evaluating classification accuracies using cross-validation. This classifier accurately predicted target identity per trial, with mean accuracies of 0.78 ± 0.13 (monkey S) and 0.83 ± 0.11 (monkey J) across all days and all learning series.

The logistic regression model weighted the contribution of each unit's firing rate across time bins between the go-cue and trial end towards classifying target identity. These weights thus represent an encoding of task information, and are related to parametric analyses of neuron activity such as direction tuning (Fig. S2C). We found that optimized classifier weights varied day to day, but became more similar as learning progressed (Fig. 2B). The correlation coefficient of the weights of stable readouts on consecutive days increased during learning (Fig. 2C), consistent with a progressive stabilization of task encoding representations that mirrors performance improvements (Fig. 1D). These findings are consistent with previous observations that adaptive BCIs lead to stable directional tuning in readout neurons [28].

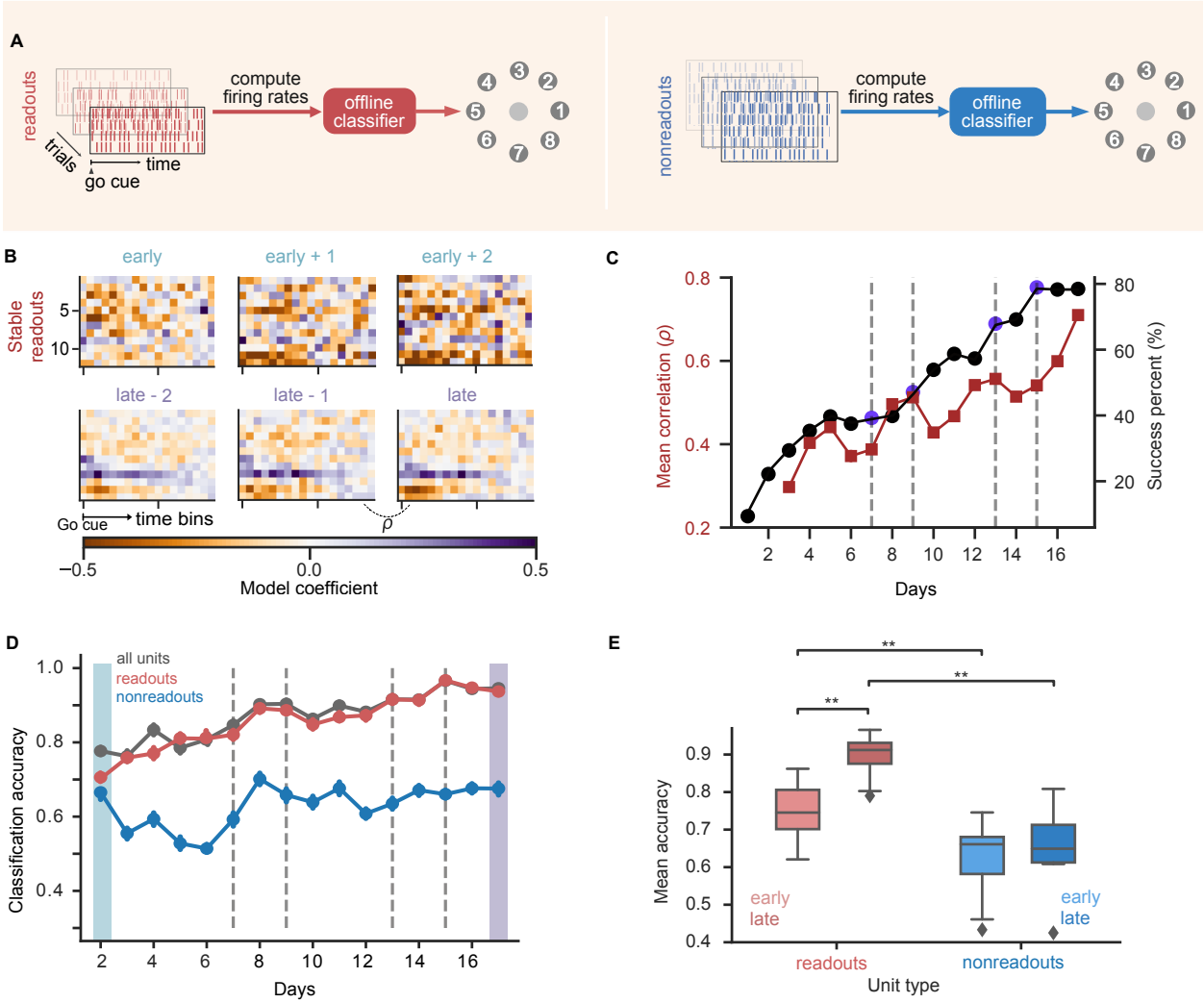

Figure 2. Credit assignment to readout units. (A) Offline classification model. **(B-D)** Classification results for the selected series (Fig. 1D). **(B)** Heatmap representation of the classifier weights to predict a single target (target 2) for the stable readout population on multiple days. **(C)** Success percentage (black; reproduced from Fig. 1D) and mean correlation coefficient between classifier weights on consecutive days (dark red) plotted across days. **(D)** Classification accuracy for all units (gray), readout units (red) and nonreadout units (blue). Error bars represent 95% confidence intervals on test accuracy (see Methods). **(E)** Early vs late classification accuracy for readout (shades of red) and nonreadout populations (shades of blue) across all series for both monkeys. (Readout-early vs readout-late : $^{**}p = 0.006$; readout-early vs nonreadout-early: $^{**}p = 0.002$; readout-late vs nonreadout-late: $^{**}p = 0.002$; nonreadout-early vs nonreadout-late: $p = 0.13$ (ns). Two-sided Wilcoxon signed-rank test, $n = 10$.)

Target classification accuracy improved during BCI training when using all neurons, paralleling the gains in BCI performance (Fig. 2D). However, this improvement was mostly driven by the readouts (Figs. 2D, E). Classification accuracy was higher for readouts compared to nonreadouts both early and late in training despite large differences in their respective numbers (Supplementary Table 1). Similar results were obtained with other classifiers and with regularization (Fig. S2A). More importantly, only the readouts' accuracy increased significantly from early to late (Fig. 2E; Fig. S2F for individual animals), showing that learning-related changes in classification accuracy are driven predominantly by changes within the readout population. We observed a similar separation in information encoding between readout and nonreadout populations when analyzing only the time period immediately after the go-cue (0-200 ms), which further controls for potential differences in reach kinematics with learning (Fig. S2G). The same analysis on data during arm movements, in contrast, showed no significant separations between readout and nonreadout populations (Fig. S2D,E), ruling out factors such as variability in neural recording quality and confirming credit assignment was specific to the BCI task [32].

Finally, we examined the temporal relationship between decoder interventions and credit assignment. We compared classification accuracies pre- and post-decoder change events and observed no significant changes (Fig. S2H). This suggests that credit assignment computations were not immediately impacted by decoder changes and instead occurred over longer timescales of practice. Together our results show that assistive decoder perturbations lead to the formation of a stable BCI task neural encoding that is primarily supported by readout units, similar to BCIs with fixed decoders [16, 32].

Compaction of neural representations with long-term learning alongside decoder adaptation

Our classifier analysis revealed a striking pattern: within a given readout ensemble, a small number of readouts carried the majority of weight for target prediction after learning (Fig. 2B, bottom row). This suggests further credit assignment within the readout population that has not been reported in past BCI studies. Does long-term learning lead certain readouts to become more important? We explored this question by conducting a rank-ordered neuron adding curve (NAC) analysis [41]. This technique quantifies how classification accuracy improves as neurons are sequentially added to a decoding model, ranked from most to least predictive (Fig. 3A).

We analyzed NACs from longitudinal BCI studies with both fixed [16] and adaptive decoders [28] to assess whether task encoding was concentrated in a few neurons or distributed across the readout population across learning. On each day (early, late), we quantified the classification accuracy of a unit in isolation (Fig. 3B,E), and then constructed a rank-ordered NAC by adding units to our decoding ensemble from most to least predictive (Fig. 3C,F). Since early fixed decoder performance was poor, we grouped data into epochs of 150 trials (early: first 150, late: last 150), consistent with previous analysis [29, 30]. Adaptive decoder datasets allowed analysis across days due to sufficient early trials. Conclusions from our analyses were not significantly impacted by the definitions of early and late datasets.

The relationship between classification accuracy and the number of ranked readout units required for prediction differed between fixed and adaptive decoders. Overall classification accuracy improved between early and late learning (Fig. S3A). We therefore examined the number of units contributing to task encoding by normalizing each day's NACs (Fig. 3C,F) and then computing the number of neurons (N_c) required to reach 80% of the normalized classification accuracy at each learning stage. In fixed decoders, late N_c exceeded early N_c, indicating broader distribution of target information with learning (Fig. 3C,D). In contrast, adaptive decoders showed the opposite trend, with a consistently smaller late N_c than early N_c (Fig. 3F,G, Fig. S3E). This effect in adaptive decoder data was even larger when including all recorded units (readouts and nonreadouts, Fig. S3B). We did not observe this effect in arm movement tasks, suggesting that the changes were specific to BCI learning and not due to recording quality (Fig. S3C). Beyond this within-series (early vs. late) trend, N_c qualitatively decreased across chronologically-ordered series for each animal (Fig. S3D). There was no clear relationship between the reduction in N_c and the number of readout units added or removed by decoder perturbations (Fig. S3F), consistent with credit assignment computations occurring on longer timescales rather than tied to any individual decoder intervention.

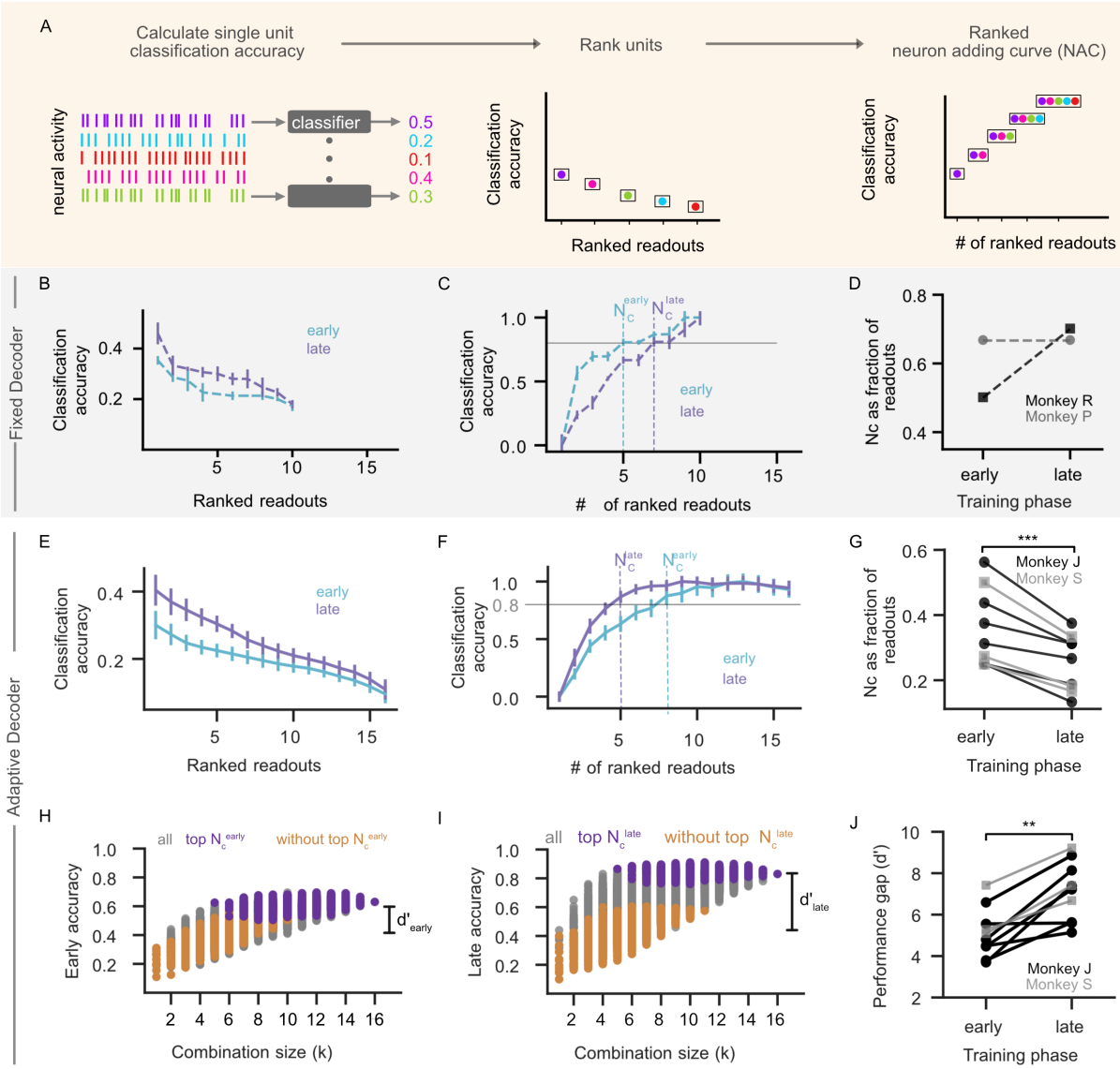

Figure 3. Learning leads to compaction of neural representations: **(A)** Ranked Neuron Adding Curve (NAC) analysis. We first estimated classification accuracies of individual units (left), ranked units based on their classification power (middle) and constructed NACs by incrementally adding the next best unit (right) (see Methods). **(B)** Ranked classification accuracy of individual units on early (cyan) and late (purple) epoch from fixed decoder dataset (example from monkey R). For the fixed decoder, early and late refer to epochs, not days, in contrast to the adaptive decoder. Epochs consist of a constant number of trials, consistent with previous analyses [30]. **(C)** Ranked NAC on early (cyan) vs late (purple) epoch for fixed decoder series from Monkey R. Accuracy is normalized to each epoch's peak performance. Vertical dashed lines indicate number of units required to reach 80% of peak accuracy (N_c). **(D)** Comparison of N_c^{late} and N_c^{early} from 2 monkeys (monkey P, grey circles; monkey R, black squares). **(E)** Same as (B) for the selected series from adaptive decoder dataset. In the adaptive decoder, early (cyan) and late (purple) refer to days, as in Fig. 1C and Fig. 2D. **(F)** Same as (C) for the selected series from adaptive decoder dataset. **(G)** Comparison of N_c^{late} and N_c^{early} as a fraction of total readouts (late < early, $^{***}p = 9.8 \times 10^{-4}$, $n = 10$, one-sided Wilcoxon signed-rank test). Monkey J: black circles; monkey S: grey squares. **(H)** Classification accuracy as a function of readout population size (combinatorial NAC) for early day. Combinations with top N_c^{early} units (deep purple); without top N_c^{early} units (orange); with a subset of top N_c^{early} units (grey). Distance between distributions with and without top N_c^{early} units defined as the discriminability index d'. **(I)** Same as (H) for late day. **(J)** Comparison of distance between distributions between early and late learning (d'_{early} vs d'_{late}) across series ($d'_{\text{late}} > d'_{\text{early}}$, $^{**}p = 1.9 \times 10^{-3}$, $n = 10$, one-sided Wilcoxon signed-rank test). Formatting as in panel G.

Next, we examined whether this reduction in N_c carried over to the encoding of kinematic variables in addition to target identity. We found that a smaller subset of readout units, identified using the classification model, was sufficient to accurately reconstruct cursor velocities offline late in learning compared to early (Fig. S3G-I and Supplementary Methods). This suggests that the same units which increasingly contributed to target encoding during learning also contributed to skilled movement. We term this reduction in the number of units needed to accurately decode behavior over time "compaction" to indicate that the brain appears to progressively encode task information in a smaller number of neural units when learning is assisted, in contrast to fixed decoder scenarios (Fig. 3D,G).

Lastly, we asked whether compaction reflected a situation where task information was encoded within a specific subset of readout neurons, or a more general increase in encoding efficiency. We performed a combinatorial variant of the neuron-adding curve analysis, in which we computed classification accuracy for all combinations of units as we vary the number of units used to decode (within readout units only). On each day, we identified the top N_c readout units and labeled combinations that contained all top units (purple) or did not contain any of the top units (orange) (Fig. 3H,I). Early and late in learning, combinations with top units held the most classification power (Fig. 3H,I). Comparing the distributions between early and late revealed that the overall increase in classification power was driven solely by the increase in classification power from top units. We quantified the performance gap between combinations with or without the top N_c^{late} units using a discriminability index (d') (see Methods). This performance gap increased with learning across series (Fig. 3J), showing that learning-related changes are targeted to a specific subset of readout neurons in BCIs with adaptive decoders.

Assistive decoder adaptation contributes to learning more compact representations: modeling analysis

Our analysis revealed that neural representations became more compact over multiple days of BCI training with assistive decoder perturbations (Fig. 3E-G), but not without them (Fig. 3B-D). However, experimental BCI data may miss potentially confounding population-level mechanisms due to practical limitations, like sparsely sampled neural activity and an inability to measure *in vivo* synaptic strengths. We therefore simulated BCI task acquisition using an artificial neural network (Fig. 4A) to directly compare learning strategies for identically initialized neural networks trained with fixed or adaptive decoders.

We developed a minimal yet biologically plausible RNN model representing a motor cortical circuit that captured the basic learning dynamics of BCI experiments (Fig. 4A). We first trained the network on a center-out arm movement task (Fig. 4B, left) to establish a plausible inductive bias for subsequent BCI training and to mirror experiments [28]. The network received context information about task type (arm vs. BCI) along with the effector and target positions. After initial training, 12 units out of 100 were selected as readouts and a velocity Kalman filter (see Methods) was fitted to manual reach trajectories (Fig. 4B, middle). The context then switched to 'BCI' and the network was trained on the BCI task (Fig. 4B, right). In all contexts the REINFORCE algorithm [42] updated recurrent (W, Fig. 4A) and input (U) weights, while the encoding (F) and output (V) matrices remained fixed after random initialization. REINFORCE does not rely on backpropagating task errors through the effectors, making it more biologically plausible.

Adaptive decoding in the model followed the principle of CLDA algorithms used in experiments [28], focusing on decoder updates when readout ensemble remained unchanged (see Methods). CLDA was performed on each "day", corresponding to seeing each target 100 times. Regardless of CLDA intensity (a factor representing how much the decoder changed; Eq. 2), adaptive decoders accelerated early performance improvements compared to fixed decoders (Fig. 4C). Moderate CLDA intensities maintained or improved final BCI performance (Fig. 4D) while preserving generalization properties comparable to fixed-decoder training (Fig. S4A). However, excessive adaptation degraded final performance (Fig. 4D, CLDA $= 0.5$), likely because CLDA's objective (reach towards the target) conflicted with the task objective (reach the target with vanishing velocity in a specified time) late in training (Methods, Eqs. 4-5). Stopping high-intensity CLDA once a performance threshold was reached mitigated this issue (Fig. S4B). Overall, in the model, decoder adaptation facilitated rapid BCI task acquisition, as in

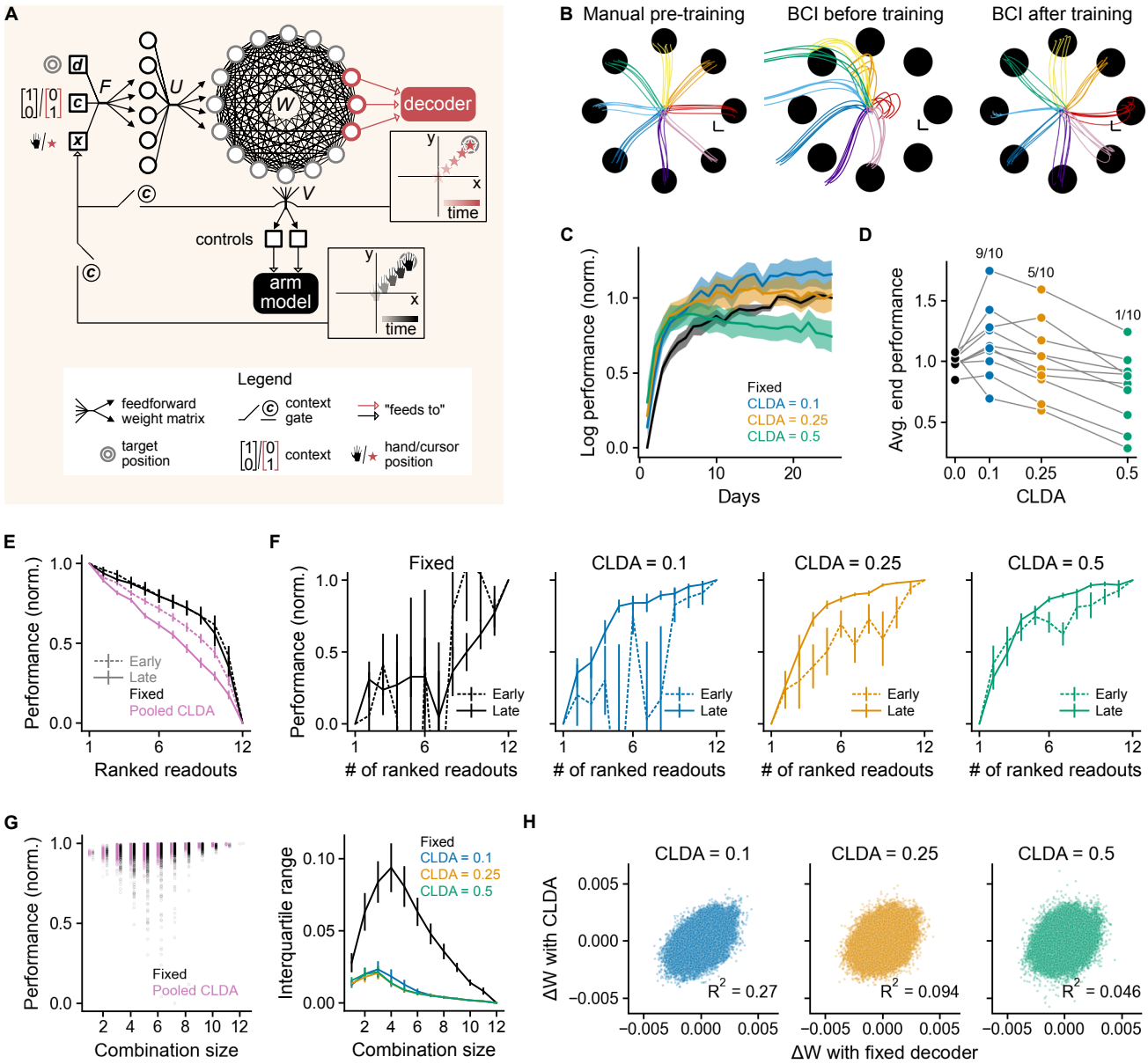

Figure 4. Decoder adaptation contributes to learning compact representations: modeling analysis **(A)** The RNN received visual feedback about target (**d**) and end-effector (**x**) positions. Based on context (**c**), it controlled either an arm model with all units or a BCI cursor with readout units (red). F, U, W and V indicate the encoding, input, recurrent and output weight matrices, respectively. See text for details. **(B)** Learning stages. Left: train on arm movement task. Middle: fit decoder parameters to arm trajectories. Right: set context to 'BCI' (here CLDA intensity = 0.25) and train on BCI task. Scale bars = 1 cm. **(C)** Normalized log performance (Methods, Eq. 7) for different CLDA intensities. **(D)** Normalized log performance averaged over last 5 days. Ratios at top are the fraction of seeds with better performance with adaptive decoders compared with fixed decoders. **(E)** Ranked single-unit normalized performance (Methods, Eq. 8). Results for adaptive decoders were pooled across CLDA intensities (pink lines). **(F)** Neuron adding curves (NACs) using online performance (Methods, Eq. 9). **(G)** Late combinatorial NAC for a given random initialization of the network parameters (left) and interquartile ranges of the distributions across realizations (right). **(H)** Change in recurrent weights (ΔW) with CLDA vs. corresponding change with a fixed decoder. In panels C, E, F and G (right), we plotted mean $\pm$ SEM, for $n = 10$ seeds, i.e. random initializations of the network.

experiments. Our results also underscore potential shortcomings of long-term CLDA applications at high intensity.

Next, we explored whether adaptive-decoder learning led to more compact representations in the model. While this was assessed offline in the data (Fig. 3), the model allowed for an online evaluation using the task loss as the performance metric (Methods, Eqs. 4-5). Each individual readout unit controlled the cursor in turn and its reach performance was evaluated both 'early' (end of first day) and 'late' (end of last day). Readout units were then ranked according to their performance. Early and late ranked single-unit performances (Methods, Eq. 8) differed only for adaptive decoders (Fig. 4E and Fig. S4C,D), with faster decrease late in training compared to early (Fig. 4E, pink lines). Thus, the dominant units became relatively more dominant with learning on average under network-decoder co-adaptation. Using these rankings, we then computed a ranked NAC (Methods, Eq. 9). Consistent with our experimental findings, adaptive decoders contributed to the compaction of representations of task performance (Fig. 4F and Fig. S4E) and kinematic variables (Fig. S4F). With a fixed decoder, ranked NACs were highly variable across seeds, with no sign of compaction as 11 (out of 12) ranked readout units were required to reach 80% of the maximal performance late in training (Fig. 4F, left). Notably, stopping CLDA early not only preserved performances (as mentioned above), but also tended to prevent the emergence of compact representations (Fig. S4G). Overall, our model confirms and expands on our observations that decoder adaptation singles out certain units for BCI control and progressively shapes more compact neural representations.

Using the online loss metric, we next obtained combinatorial unit-adding curves (Fig. 4G, left) and computed the corresponding interquartile ranges (IQR; Fig. 4G, right). These IQRs measured the dispersion of performance for each combination size (Methods, Eq. 10). The IQRs were significantly greater for a fixed decoder, which was caused by few units with very low contributions to the overall performance (Fig. 4E and Fig. S4C). Therefore, while decoder adaptation promoted more compact representations with a few dominant units, it also mitigated the impact of unreliable ones.

The model suggests that neural plasticity alone, without decoder adaptation, does not lead to compaction. Thus, decoder adaptation must interact with the model's plasticity to generate this feature of neural representations. To show this, we calculated the total changes in recurrent weights across learning, ΔW, with and without decoder adaptation, and computed their coefficients of determination (Fig. 4H). All connections—among and across readout and nonreadout units—were included. Since our simulations were precisely matched in terms of random number generation across CLDA intensities, comparing these weight changes one-to-one was meaningful. The coefficients of determination decreased as CLDA intensity increased, indicating that the total weight changes with an adaptive decoder became increasingly uncorrelated from those obtained with a fixed decoder. These results suggest that brain-decoder co-adaptation drives distinct neural plasticity patterns and contributes to compaction.

Compact representations encode task information in low-variance population modes

Our neural data analysis and model revealed that compaction at the unit level in neural representations emerge as the result of long-term learning processes occurring alongside an adaptive decoder. Does this reshaping of sensorimotor encoding at the level of individual neurons reflect a broader reorganization within the neural population? Many studies represent coordinated neural activity with principal components (PCs) capturing the most prominent directions of variability [43, 44, 45]. In this view, each PC can be treated as an independent component or "mode" of neural population activity.

We explored how compaction changed population structure by performing a variation of our neuron-adding curve analysis in which population-level modes served as features for classification rather than individual units (Fig. 5A). For early and late days, we performed principal component analysis, ranked PCs based on their classification accuracy (Fig. 5B) and computed a rank-ordered PC adding curve (Fig. 5C). Similar to the compaction observed at the unit level, we found that the number of PCs required for 80% classification accuracy decreased with learning (Fig. 5C,D). This observation is not a trivial consequence of unit-level compaction (see Supplementary Discussion), and confirms compaction of representations at the level of populations.

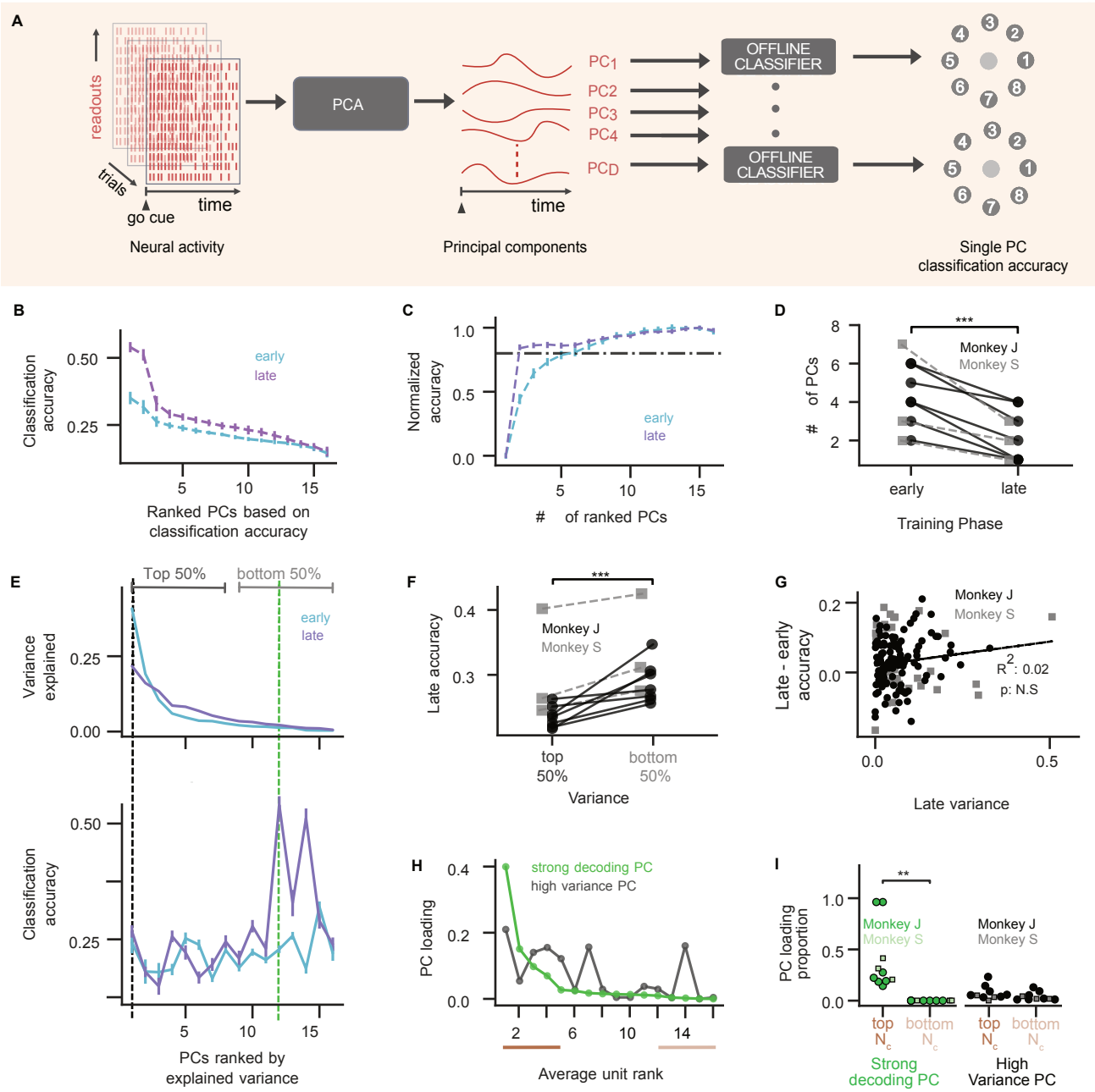

Figure 5. Task information emerges in low-variance modes **(A)** Decoding analysis with PC modes (see text). **(B)** Individual PC classification accuracy (similar to Fig. 3E) for early (cyan) and late (purple) learning phases from selected series. PCs ranked by classification accuracy. **(C)** Ranked dimension adding curve (similar to ranked NAC in Fig. 3F) for early and late days for the example series. **(D)** Comparison of number of PCs needed for 80% classification accuracy on early and late days across series ($^{***}p = 9.7 \times 10^{-4}$, $n = 10$, late $<$ early, one-sided Wilcoxon signed-rank test). Monkey J: black circles; monkey S: grey squares. **(E)** Top: Variance explained by each PC (ranked by variance explained). Bottom: Classification accuracy per PC, ordered by variance explained, Error bars denote 95% confidence intervals from 10^4 random draws of trials. The strongly decoding PC (green) has the highest accuracy. **(F)** Comparison of mean classification accuracy from top 50% vs bottom 50% PCs during late learning day across series. Top 50% is lower than bottom 50%: $^{**}p = 0.001$, $n = 10$, one-sided Wilcoxon signed-rank test. Data formatting as in D. **(G)** Change in classification accuracy (late $-$ early) plotted versus late variance per PC. No significant trend: $R^2 = 0.01$, $p = 0.26$, $n = 10$. Data formatting as in D. **(H)** Square of PC loadings from each readout unit for the strong decoding PC (green) and high variance PC (black). Units on the x-axis are ranked by their average classification performance. **(I)** Comparison of sum of squares of PC loadings from top 4 ranked readouts and bottom 4 ranked readouts from strong decoding PC and high variance PC across all series. Strong decoding PC: $^{**}p = 0.002$; high variance PC: $p = 0.38$ (ns), two-sided Wilcoxon signed-rank test, $n = 10$.

We further observed that variance became more broadly distributed across PCs in the readout population after learning, suggesting an increase in dimensionality (Fig. 5E, top). To validate this observation, we calculated the dimensionality of readout activity using the participation ratio, a covariance-based dimensionality measure [46, 47], normalized by the number of readouts (PR_{norm}). Unlike in natural motor learning studies where neural variability tends to shrink [31] or fixed-decoder BCI studies where "shared" dimensionality decreases [29, 30], dimensionality remained the same or increased during BCI learning with adaptive decoders (Fig. S5A,C). Re-analyzing fixed-decoder data using PR_{norm} confirmed that dimensionality changes with learning differ in the presence of adaptive decoders (Fig. S5B,C).

How can neural representations become more compact while dimensionality does not decrease? We linked variance captured by population-level modes with task information across learning. While variance was more distributed across modes after learning (Fig. 5E, top), modes capturing the highest population variance were not necessarily the most predictive (Fig. 5E, bottom). We quantified the relationship between variance explained and task information for population modes by dividing PCs into two groups for each learning series: the top 50%, which explained approximately 80-84% of population variance, and the bottom 50%, which explained 16-22% of population variance. Across all learning series, the average classification accuracy on late days was higher in the bottom 50% group compared to the top 50% group (Fig. 5F), showing that task information was more often found in low-variance modes after learning. Moreover, the changes in a PC's classification accuracy during learning were not correlated with its variance explained (Fig. 5G). Thus, compaction co-occurred with a nondecreasing dimensionality because task-relevant information became embedded in the low-variance modes.

Lastly, we examined whether, after learning, task-predictive units contributed preferentially to task-predictive population modes. We ranked readouts based on their single-unit predictive power and then quantified their influence on PC modes using the squared PC loadings (see Methods). Comparing PC loadings for an example high-variance PC with those of a strongly decoding PC revealed starkly different patterns of unit contributions (Fig. 5H). As expected, strongly decoding PCs receive large contributions from strongly decoding units. In contrast, high-variance PCs have contributions from a seemingly random mix of units. To quantify this effect across all learning series, we compared the sum of squared PC loadings from the top N_c task-predictive units to the bottom N_c task-predictive units for the strongest decoding PC and highest variance PC in each series (Fig. 5I). Strongly decoding PCs consistently drew more from the top-performing units, with negligible input from the least predictive units, a pattern not observed in high-variance PCs (Fig. 5I). Thus, the most predictive PCs in neural representations learned with adaptive decoders were not only associated with moderate to low variance but also composed primarily of the most task-predictive units. This indicates that task-relevant information can be compactly encoded in low-variance population modes through the coordinated contributions of highly informative units, challenging conventional views on the relationship between neural variability and task encoding.

Discussion

We found that assistive BCI perturbations led to a compaction of neural representations, concentrating task information into fewer neurons and neural modes that capture a relatively small portion of the total population variance. Both experimental data and our model suggest that compaction does not occur without these assistive perturbations. By revealing new impacts of BCI *decoders* on the brain's *encoder*, our work also opens new opportunities to design BCI algorithms that deliberately steer a two-learner system toward better task representations.

Our findings highlight the complexity of "error" signals in BCI learning. Past BCI studies emphasized links between neural changes and task performance. Neural activity may only reorganize when decoder perturbations introduce task errors and violate existing neural correlations [15, 48]; and adapting a decoder to decrease task errors reduces neural changes over time [28]. Unlike most prior studies [14, 15, 35, 48, 49], here the decoder changes did not introduce overt *task* errors, instead creating intermittent changes to the mapping between neural activity and behavior that do not always have an impact on task performance. Adaptive decoders, therefore, manipulate the mapping between *intent* and *control*, even when task error is not affected. We

found that these decoder changes impacted learned neural representations (experiment, Fig. 3B-G and model, Fig. 4E-G) and synaptic-level changes (model; Fig. 4H). This is consistent with our hypothesis that decoder adaptation interacts with error-driven skill learning computations, and suggests that learned representations in BCI tasks are influenced by error signals beyond that of task objectives. Motor learning studies suggest that skill acquisition involves both fast (within-session) and slow (multi-day) adaptation [1, 2, 35, 50] with different error sensitivities [51]. Fast and slow adaptation likely involve different neural changes, such as fast adaptation via minimal synaptic changes [49, 52]. One possibility that can be explored in future studies is that the initial performance boost from adaptive decoders could supplant rapid learning mechanisms driven by task errors, while slow mechanisms that shape synaptic changes remain and are impacted by the algorithm. This is consistent with our observation that compaction emerged over time, rather than explicitly tied to individual decoder update events (Fig. S2H, S3F), and trends of more compact representations *across* learning series (Fig. S3D). Indeed, adaptive decoders may alter slower credit-assignment processes using error signals generated through mechanisms like an “internal model" [53, 54].

Our results also indicate that credit-assignment computations during extended BCI practice operate at scales spanning single neurons as well as coordinated neural populations, and that these computations are impacted by adaptive decoders. Importantly, compaction of population modes and that of units do not trivially follow one another in general (see also Supplementary Discussion), making it noteworthy that dominant units preferentially contribute to strongly-decoding modes (Fig. 5H-I). Our offline classification of target identity with neural data (Fig. 3, Fig. 5) and online assessment of neuronal contributions to control in the model (Fig. 4) revealed that adaptive decoders strongly influence which neurons or neural modes participate in the task, leading to exaggerated credit assignment. These task informative neurons appear to form an exclusive coordinated population, since task informative population modes have minimal contributions from other neurons. This exclusion, in turn, partially explains why modes with task information ultimately capture relatively little of the overall population variance. This is consistent with our supplemental analysis comparing variance and dimensionality changes during learning between fixed and adaptive decoders. Neural variability and dimensionality have been found to reduce with practice (along with behavioral variability) in both motor learning [31] and BCI learning with fixed decoders [29, 30]. We found that this is not the case when the decoder adapts, instead leaving population dimensionality and variability largely unchanged (Fig. S5BC). Together, these results suggest that assistive decoders may shape neuron-level learning processes in a way that helps maintain population dimensionality. Our study adds to observations that neural variance and task information are not necessarily linked [55], showing that long-term learning phenomena can occur in relatively low-variance modes where they may easily be missed by linear-based variance analyses [40].

Our study reveals new challenges and opportunities in designing long-term BCIs. Given measurement instabilities over a BCI’s lifetime, some form of adaptive decoding is unavoidable. Our results highlight that long-term learning mechanisms will likely occur alongside decoder changes, making long-term BCI systems inherently *co-adaptive* [25, 56]. Moreover, our results demonstrate the inherent entangling of *decoding* and *encoding* in co-adaptive systems. The adaptive decoding algorithms used in our experiments [20]—closely related to widely used adaptive algorithms [11, 21, 23, 24, 26, 36, 37]—produced compaction of neural representations. This suggests that the brain-decoder system ultimately relies on a smaller set of neural features, which will impact BCI performance. For instance, compact representations may improve computational efficiency, and could lead to task encodings that are more easily separated from other behaviors to reduce interference from other task demands [28]. Compact encoder-decoder systems may also protect against unreliable units (Fig. 4G). Yet, the dominance of a few features can prevent more subtle information from influencing learning [57] and compact task encodings may increase sensitivity to neural recording loss. Our model highlights the opportunity to balance these potential trade-offs by properly fine-tuning the decoder adaptation algorithm. Such models will likely be valuable for more principled designs of co-adaptive BCIs [56]. Understanding the mechanisms by which assistive decoders shape task encoding and their functional implications will ultimately make it possible to design BCIs that provide high performance for someone’s lifetime.

Acknowledgements

The authors thank Jose M. Carmena, who shared data collected in his laboratory for this study, and Vivek Athalye for his assistance with the fixed decoder dataset. The authors' research was supported in part by an IVADO postdoctoral fellowship, Canada First Research Excellence Fund/Apogée (AP), an NSF Accelnet INBIC fellowship (PR), a Simons Collaboration for the Global Brain Pilot award (898220, GL and ALO), a Google Faculty Award (ALO and GL), an NSERC Discovery Grant (RGPIN-2018-04821, GL), the Eunice Kennedy Shriver National Institute of Child Health and Human Development (NIH K12HD073945, ALO), and NIH grant R01 NS134634 (ALO). GL further acknowledges support from the Canada CIFAR AI Chair Program and the Canada Research Chair in Neural Computations and Interfacing (CIHR, tier 2). The content of the present paper is solely the responsibility of the authors and does not necessarily represent the official views of the funding agencies.

Declaration of Interests

A.L.O. is a scientific advisor for Meta Reality Labs. G.L. is a scientific advisor for BIOS Health, Neural Drive, Neurosoft Bioelectronics as well as a Visiting Faculty Researcher at Google.

References

[1] John W. Krakauer et al. "Motor Learning". en. In: *Comprehensive Physiology*. John Wiley & Sons, Ltd, 2019, pp. 613–663. ISBN: 978-0-470-65071-4.

[2] Eran Dayan and Leonardo G. Cohen. "Neuroplasticity Subserving Motor Skill Learning". en. In: *Neuron* 72.3 (Nov. 2011), pp. 443–454. ISSN: 0896-6273. DOI: 10.1016/j.neuron.2011.10.008.

[3] Ling Wang et al. "Structural plasticity within highly specific neuronal populations identifies a unique parcellation of motor learning in the adult brain". In: *Proceedings of the National Academy of Sciences* 108.6 (Feb. 2011). Publisher: Proceedings of the National Academy of Sciences, pp. 2545–2550. DOI: 10.1073/pnas.1014335108.

[4] Tonghui Xu et al. "Rapid formation and selective stabilization of synapses for enduring motor memories". en. In: *Nature* 462.7275 (Dec. 2009). Number: 7275 Publisher: Nature Publishing Group, pp. 915–919. ISSN: 1476-4687. DOI: 10.1038/nature08389.

[5] Min Fu et al. "Repetitive motor learning induces coordinated formation of clustered dendritic spines in vivo". en. In: *Nature* 483.7387 (Mar. 2012). Number: 7387 Publisher: Nature Publishing Group, pp. 92–95. ISSN: 1476-4687. DOI: 10.1038/nature10844.

[6] Blake A Richards and Timothy P Lillicrap. "Dendritic solutions to the credit assignment problem". en. In: *Current Opinion in Neurobiology*. Neurobiology of Learning and Plasticity 54 (Feb. 2019), pp. 28–36. ISSN: 0959-4388. DOI: 10.1016/j.conb.2018.08.003.

[7] Matthew D Golub et al. "Brain–computer interfaces for dissecting cognitive processes underlying sensorimotor control". In: *Current Opinion in Neurobiology*. Neurobiology of cognitive behavior 37 (Apr. 2016), pp. 53–58. ISSN: 0959-4388. DOI: 10.1016/j.conb.2015.12.005.

[8] Amy L. Orsborn and Bijan Pesaran. "Parsing learning in networks using brain-machine interfaces". eng. In: *Current Opinion in Neurobiology* 46 (Oct. 2017), pp. 76–83. ISSN: 1873-6882. DOI: 10.1016/j.conb.2017.08.002.

[9] John K. Chapin et al. "Real-time control of a robot arm using simultaneously recorded neurons in the motor cortex". en. In: *Nature Neuroscience* 2.7 (July 1999). Number: 7 Publisher: Nature Publishing Group, pp. 664–670. ISSN: 1546-1726. DOI: 10.1038/10223.

[10] Jose M. Carmena et al. "Learning to Control a Brain–Machine Interface for Reaching and Grasping by Primates". en. In: *PLOS Biology* 1.2 (Oct. 2003). Publisher: Public Library of Science, e42. ISSN: 1545-7885. DOI: 10.1371/journal.pbio.0000042.

[11] Dawn M. Taylor, Stephen I. Helms Tillery, and Andrew B. Schwartz. "Direct Cortical Control of 3D Neuroprosthetic Devices". In: *Science* 296.5574 (June 2002). Publisher: American Association for the Advancement of Science, pp. 1829–1832. DOI: 10.1126/science.1070291.

[12] R. Shadmehr and F. A. Mussa-Ivaldi. "Adaptive representation of dynamics during learning of a motor task". In: *The Journal of Neuroscience: The Official Journal of the Society for Neuroscience* 14.5 (May 1994), pp. 3208–3224. ISSN: 0270-6474. DOI: 10.1523/JNEUROSCI.14-05-03208.1994.

[13] John W. Krakauer et al. "Learning of Visuomotor Transformations for Vectorial Planning of Reaching Trajectories". In: *Journal of Neuroscience* 20.23 (Dec. 1, 2000). Publisher: Society for Neuroscience Section: ARTICLE, pp. 8916–8924. ISSN: 0270-6474, 1529-2401. DOI: 10.1523/JNEUROSCI.20-23-08916.2000.

[14] Beata Jarosiewicz et al. "Functional network reorganization during learning in a brain-computer interface paradigm". In: *Proceedings of the National Academy of Sciences of the United States of America* 105.49 (Dec. 2008), pp. 19486–19491. ISSN: 0027-8424. DOI: 10.1073/pnas.0808113105.

[15] Patrick T. Sadtler et al. "Neural constraints on learning". en. In: *Nature* 512.7515 (Aug. 2014). Number: 7515 Publisher: Nature Publishing Group, pp. 423–426. ISSN: 1476-4687. DOI: 10.1038/nature13665.

[16] Karunesh Ganguly and Jose M. Carmena. "Emergence of a Stable Cortical Map for Neuroprosthetic Control". en. In: *PLOS Biology* 7.7 (July 2009). Publisher: Public Library of Science, e1000153. ISSN: 1545-7885. DOI: 10.1371/journal.pbio.1000153.

[17] Aaron C. Koralek et al. "Corticostriatal plasticity is necessary for learning intentional neuroprosthetic skills". en. In: *Nature* 483.7389 (Mar. 2012). Number: 7389 Publisher: Nature Publishing Group, pp. 331–335. ISSN: 1476-4687. DOI: 10.1038/nature10845.

[18] Vivek R Athalye, Jose M Carmena, and Rui M Costa. "Neural reinforcement: re-entering and refining neural dynamics leading to desirable outcomes". en. In: *Current Opinion in Neurobiology*. Neurobiology of Behavior 60 (Feb. 2020), pp. 145–154. ISSN: 0959-4388. DOI: 10.1016/j.conb.2019.11.023.

[19] Amy Orsborn and Jose Carmena. "Creating new functional circuits for action via brain-machine interfaces". In: *Frontiers in Computational Neuroscience* 7 (2013). ISSN: 1662-5188.

[20] Amy L. Orsborn et al. "Closed-loop decoder adaptation on intermediate time-scales facilitates rapid BMI performance improvements independent of decoder initialization conditions". eng. In: *IEEE transactions on neural systems and rehabilitation engineering: a publication of the IEEE Engineering in Medicine and Biology Society* 20.4 (July 2012), pp. 468–477. ISSN: 1558-0210. DOI: 10.1109/TNSRE.2012.2185066.

[21] Alim Louis Benabid et al. "An exoskeleton controlled by an epidural wireless brain-machine interface in a tetraplegic patient: a proof-of-concept demonstration". In: *The Lancet. Neurology* 18.12 (2019), pp. 1112–1122. ISSN: 1474-4465. DOI: 10.1016/S1474-4422(19)30321-7.

[22] Zheng Li et al. "Adaptive Decoding for Brain-Machine Interfaces Through Bayesian Parameter Updates". In: *Neural Computation* 23.12 (Dec. 2011), pp. 3162–3204. ISSN: 0899-7667. DOI: 10.1162/NECO_a_00207.

[23] Vikash Gilja et al. "A high-performance neural prosthesis enabled by control algorithm design". en. In: *Nature Neuroscience* 15.12 (Dec. 2012), pp. 1752–1757. ISSN: 1546-1726. DOI: 10.1038/nn.3265.

[24] Vikash Gilja et al. "Clinical translation of a high-performance neural prosthesis". In: *Nature Medicine* 21.10 (Sept. 28, 2015), pp. 1142–1145. ISSN: 1078-8956, 1546-170X. DOI: 10.1038/nm.3953.

[25] Krishna V. Shenoy and Jose M. Carmena. "Combining Decoder Design and Neural Adaptation in Brain-Machine Interfaces". en. In: *Neuron* 84.4 (Nov. 2014), pp. 665–680. ISSN: 0896-6273. DOI: 10.1016/j.neuron.2014.08.038.

[26] Beata Jarosiewicz et al. "Virtual typing by people with tetraplegia using a self-calibrating intracortical brain-computer interface". In: *Science Translational Medicine* 7.313 (Nov. 2015). Publisher: American Association for the Advancement of Science, 313ra179–313ra179. DOI: 10.1126/scitranslmed.aac7328.

[27] Chethan Pandarinath et al. “High performance communication by people with paralysis using an intracortical brain-computer interface”. In: *eLife* 6 (Feb. 2017). Ed. by Sabine Kastner. Publisher: eLife Sciences Publications, Ltd, e18554. ISSN: 2050-084X. DOI: 10.7554/eLife.18554.

[28] Amy L. Orsborn et al. “Closed-Loop Decoder Adaptation Shapes Neural Plasticity for Skillful Neuroprosthetic Control”. en. In: *Neuron* 82.6 (June 2014), pp. 1380–1393. ISSN: 0896-6273. DOI: 10.1016/j.neuron.2014.04.048.

[29] Vivek R. Athalye et al. “Emergence of Coordinated Neural Dynamics Underlies Neuroprosthetic Learning and Skillful Control”. eng. In: *Neuron* 93.4 (Feb. 2017), 955–970.e5. ISSN: 1097-4199. DOI: 10.1016/j.neuron.2017.01.016.

[30] Ellen L. Zippi et al. *Selective modulation of population dynamics during neuroprosthetic skill learning*. en. Pages: 2021.01.08.425917 Section: New Results. Jan. 2021. DOI: 10.1101/2021.01.08.425917.

[31] Ashesh K. Dhawale, Maurice A. Smith, and Bence P. Ölveczky. “The Role of Variability in Motor Learning”. In: *Annual Review of Neuroscience* 40.1 (2017). _eprint: https://doi.org/10.1146/annurev-neuro-072116-031548, pp. 479–498. DOI: 10.1146/annurev-neuro-072116-031548.

[32] Karunesh Ganguly et al. “Reversible large-scale modification of cortical networks during neuroprosthetic control”. en. In: *Nature Neuroscience* 14.5 (May 2011). Number: 5 Publisher: Nature Publishing Group, pp. 662–667. ISSN: 1546-1726. DOI: 10.1038/nn.2797.

[33] Tanuj Gulati et al. “Neural reactivations during sleep determine network credit assignment”. en. In: *Nature Neuroscience* 20.9 (Sept. 2017). Number: 9 Publisher: Nature Publishing Group, pp. 1277–1284. ISSN: 1546-1726. DOI: 10.1038/nn.4601.

[34] Pierre-Jean Arduin et al. ““Master” Neurons Induced by Operant Conditioning in Rat Motor Cortex during a Brain-Machine Interface Task”. en. In: *Journal of Neuroscience* 33.19 (May 2013). Publisher: Society for Neuroscience Section: Articles, pp. 8308–8320. ISSN: 0270-6474, 1529-2401. DOI: 10.1523/JNEUROSCI.2744-12.2013.

[35] Xiao Zhou et al. “Distinct types of neural reorganization during long-term learning”. In: *Journal of Neurophysiology* 121.4 (Apr. 2019). Publisher: American Physiological Society, pp. 1329–1341. ISSN: 0022-3077. DOI: 10.1152/jn.00466.2018.

[36] Daniel B. Silversmith et al. “Plug-and-play control of a brain–computer interface through neural map stabilization”. en. In: *Nature Biotechnology* 39.3 (Mar. 2021). Number: 3 Publisher: Nature Publishing Group, pp. 326–335. ISSN: 1546-1696. DOI: 10.1038/s41587-020-0662-5.

[37] Jennifer L Collinger et al. “High-performance neuroprosthetic control by an individual with tetraplegia”. In: *The Lancet* 381.9866 (Feb. 2013), pp. 557–564. ISSN: 01406736. DOI: 10.1016/S0140-6736(12)61816-9.

[38] B. Wodlinger et al. “Ten-dimensional anthropomorphic arm control in a human brain-machine interface: difficulties, solutions, and limitations”. eng. In: *Journal of Neural Engineering* 12.1 (Feb. 2015), p. 016011. ISSN: 1741-2552. DOI: 10.1088/1741-2560/12/1/016011.

[39] Andres Agudelo-Toro et al. “Accurate neural control of a hand prosthesis by posture-related activity in the primate grasping circuit”. In: *Neuron* 112.24 (Dec. 2024), 4115–4129.e8. ISSN: 0896-6273. DOI: 10.1016/j.neuron.2024.09.018.

[40] Eva L. Dyer and Konrad Kording. “Why the simplest explanation isn’t always the best”. In: *Proceedings of the National Academy of Sciences* 120.52 (Dec. 2023). Publisher: Proceedings of the National Academy of Sciences, e2319169120. DOI: 10.1073/pnas.2319169120.

[41] Mikhail A. Lebedev. “How to read neuron-dropping curves?” In: *Frontiers in Systems Neuroscience* 8 (May 2014). ISSN: 1662-5137. DOI: 10.3389/fnsys.2014.00102.

[42] Ronald J. Williams. “Simple statistical gradient-following algorithms for connectionist reinforcement learning”. en. In: *Machine Learning* 8.3 (May 1992), pp. 229–256. ISSN: 1573-0565. DOI: 10.1007/BF00992696.

[43] John P Cunningham and Byron M Yu. "Dimensionality reduction for large-scale neural recordings". en. In: *Nature Neuroscience* 17.11 (Nov. 2014), pp. 1500–1509. ISSN: 1097-6256, 1546-1726. DOI: 10.1038/nn.3776.

[44] Chethan Pandarinath et al. "Inferring single-trial neural population dynamics using sequential auto-encoders". In: *Nature Methods* 15.10 (Oct. 2018), pp. 805–815. ISSN: 1548-7105. DOI: 10.1038/s41592-018-0109-9.

[45] Juan A. Gallego et al. "Cortical population activity within a preserved neural manifold underlies multiple motor behaviors". en. In: *Nature Communications* 9.1 (Oct. 2018). Number: 1 Publisher: Nature Publishing Group, p. 4233. ISSN: 2041-1723. DOI: 10.1038/s41467-018-06560-z.

[46] Stefano Recanatesi et al. "Dimensionality in recurrent spiking networks: Global trends in activity and local origins in connectivity". In: *PLoS Computational Biology* 15.7 (July 2019), e1006446. ISSN: 1553-734X. DOI: 10.1371/journal.pcbi.1006446.

[47] Peiran Gao et al. *A theory of multineuronal dimensionality, dynamics and measurement*. en. Tech. rep. Company: Cold Spring Harbor Laboratory Distributor: Cold Spring Harbor Laboratory Label: Cold Spring Harbor Laboratory Section: New Results Type: article. Nov. 2017, p. 214262. DOI: 10.1101/214262.

[48] Emily R. Oby et al. "New neural activity patterns emerge with long-term learning". en. In: *Proceedings of the National Academy of Sciences* 116.30 (July 2019), pp. 15210–15215. ISSN: 0027-8424, 1091-6490. DOI: 10.1073/pnas.1820296116.

[49] Matthew D. Golub et al. "Learning by neural reassociation". en. In: *Nature Neuroscience* 21.4 (Apr. 2018), pp. 607–616. ISSN: 1097-6256, 1546-1726. DOI: 10.1038/s41593-018-0095-3.

[50] Avi Karni et al. "The acquisition of skilled motor performance: Fast and slow experience-driven changes in primary motor cortex". In: *Proceedings of the National Academy of Sciences* 95.3 (Feb. 1998). Publisher: Proceedings of the National Academy of Sciences, pp. 861–868. DOI: 10.1073/pnas.95.3.861.

[51] Maurice A. Smith, Ali Ghazizadeh, and Reza Shadmehr. "Interacting Adaptive Processes with Different Timescales Underlie Short-Term Motor Learning". en. In: *PLOS Biology* 4.6 (May 2006). Publisher: Public Library of Science, e179. ISSN: 1545-7885. DOI: 10.1371/journal.pbio.0040179.

[52] Matthew G. Perich, Juan A. Gallego, and Lee E. Miller. "A Neural Population Mechanism for Rapid Learning". en. In: *Neuron* 100.4 (Nov. 2018), 964–976.e7. ISSN: 0896-6273. DOI: 10.1016/j.neuron.2018.09.030.

[53] Daniel M. Wolpert and J. Randall Flanagan. "Motor prediction". English. In: *Current Biology* 11.18 (Sept. 2001). Publisher: Elsevier, R729–R732. ISSN: 0960-9822. DOI: 10.1016/S0960-9822(01)00432-8.

[54] Matthew D Golub, Byron M Yu, and Steven M Chase. "Internal models for interpreting neural population activity during sensorimotor control". In: *eLife* 4 (Dec. 2015). Ed. by Timothy Behrens. Publisher: eLife Sciences Publications, Ltd, e10015. ISSN: 2050-084X. DOI: 10.7554/eLife.10015.

[55] Abigail A. Russo et al. "Motor cortex embeds muscle-like commands in an untangled population response". In: *Neuron* 97.4 (Feb. 2018), 953–966.e8. ISSN: 0896-6273. DOI: 10.1016/j.neuron.2018.01.004.

[56] Maneeshika M. Madduri, Samuel A. Burden, and Amy L. Orsborn. "Biosignal-based co-adaptive user-machine interfaces for motor control". In: *Current Opinion in Biomedical Engineering* 27 (Sept. 2023), p. 100462. ISSN: 2468-4511. DOI: 10.1016/j.cobme.2023.100462.

[57] Mohammad Pezeshki et al. "Gradient Starvation: A Learning Proclivity in Neural Networks". In: *Advances in Neural Information Processing Systems*. Vol. 34. Curran Associates, Inc., 2021, pp. 1256–1272.

[58] R. E. Kalman. "A New Approach to Linear Filtering and Prediction Problems". In: *Journal of Basic Engineering* 82.1 (Mar. 1960), pp. 35–45. ISSN: 0021-9223. DOI: 10.1115/1.3662552.

[59] *sklearn.linear_model.LogisticRegression — scikit-learn 0.24.1 documentation*.

[60] I.T. Jolliffe. *Principal Component Analysis*. Second edition. Springer Series in Statistics. New York, NY: Springer, 2002. ISBN: 978-0-387-95442-4.

[61] Gaël Guennebaud, Benoît Jacob, et al. *Eigen v3*. http://eigen.tuxfamily.org. 2010.

[62] Weiwei Li and Emanuel Todorov. “Iterative linear quadratic regulator design for nonlinear biological movements systems”. en. In: *Proceedings of the First International Conference on Informatics in Control, Automation and Robotics*. 2004, pp. 222–229. ISBN: 978-972-8865-12-2. DOI: 10.5220/0001143902220229.

[63] Emanuel Todorov. “Optimal Control Theory”. In: *Bayesian Brain: Probabilistic Approaches to Neural Coding*. _eprint: https://direct.mit.edu/book/chapter-pdf/2321908/9780262294188_cal.pdf. The MIT Press, Dec. 2006. ISBN: 978-0-262-29418-8. DOI: 10.7551/mitpress/1535.003.0018.

[64] Ernest J. Cheng and Stephen H. Scott. “Morphometry of Macaca mulatta forelimb. I. Shoulder and elbow muscles and segment inertial parameters”. en. In: *Journal of Morphology* 245.3 (2000), pp. 206–224. ISSN: 1097-4687. DOI: https://doi.org/10.1002/1097-4687(200009)245:3<206::AID-JMOR3>3.0.CO;2-U.

[65] Emanuel Todorov. “Direct cortical control of muscle activation in voluntary arm movements: a model”. en. In: *Nature Neuroscience* 3.4 (Apr. 2000), pp. 391–398. ISSN: 1546-1726. DOI: 10.1038/73964.

[66] Jiahao Chen and Hong Qiao. “Motor-Cortex-Like Recurrent Neural Network and Multi-Tasks Learning for the Control of Musculoskeletal Systems”. In: *IEEE Transactions on Cognitive and Developmental Systems* (2020), pp. 1–1. ISSN: 2379-8939. DOI: 10.1109/TCDS.2020.3045574.

[67] Nicolas Frémaux, Henning Sprekeler, and Wulfram Gerstner. “Functional Requirements for Reward-Modulated Spike-Timing-Dependent Plasticity”. en. In: *Journal of Neuroscience* 30.40 (Oct. 2010), pp. 13326–13337. ISSN: 0270-6474, 1529-2401. DOI: 10.1523/JNEUROSCI.6249-09.2010.

[68] Diederik P. Kingma and Jimmy Ba. “Adam: A Method for Stochastic Optimization”. In: *arXiv:1412.6980 [cs]* (Jan. 2017). arXiv: 1412.6980.

Methods

Experiment

Neural data and behavioral task

We analyzed neural activity as two male rhesus macaques (*Macaca mulatta*, monkeys S and J) learned to move a 2D cursor controlled via neural activity to perform a point-to-point reaching task while the relationship between neural activity and cursor movement was intermittently adapted across days (Fig.1C). The task required animals to initiate a trial by moving the cursor to the central target. After reaching the center, one of eight peripheral targets appeared. After a brief hold at the center, a target color change cued the animal to start reaching. Successfully completing a trial required the monkey to enter the peripheral target within a specified reach time ($3-10$ s) and hold the cursor inside the target briefly ($250-400$ ms). The monkey then received a liquid reward. Because we were primarily interested in neural representations of movement direction, we analyzed all trials in which the animal successfully reached the peripheral target regardless of whether the peripheral hold was successful or not.

Both monkeys were implanted with 128 microwire electrode arrays in motor and premotor cortices to record spiking activity. Spike-sorted multi-unit activity (monkey S) or unsorted threshold crossings defining channel-level multi-unit activity (monkey J) were used for BCI control. We refer to neural activity for both animals as “units”. A subset of recorded units was used for the real-time BCI cursor control, which defines the readout neural population. Readout neurons were chosen based primarily on their stability across days, assessed qualitatively by experimenters by observing recordings over time. Functional properties, such as tuning for arm movements or offline task predictive power, were not used to inform readout unit selection. Units that were recorded on the same array, but not used as inputs to the BCI decoder for cursor control, define the nonreadout neural population. Stable readout units were defined as units consistently used throughout an entire BCI series.

All procedures were conducted in compliance with the NIH Guide for the Care and Use of Laboratory Animals and were approved by the University of California at Berkeley Institutional Animal Care and Use Committee.

BCI control with adaptive decoders

Readout neural activity controlled the 2D cursor using a position-velocity Kalman filter (KF) [20, 23, 28, 58]. KF parameters were trained and intermittently updated using previously-developed algorithms that estimate parameters during closed-loop BCI control (closed-loop decoder adaptation, CLDA) [20]. CLDA re-estimates the parameters of the KF using the subject's neural activity and estimated motor intent during BCI control and updates the decoder used for BCI control in real-time. Updates to the KF BCI used the SmoothBatch algorithm [20] (monkey J; see below) which updates a subset of the KF parameters or the ReFIT algorithm [23] (monkey S) which updates all KF parameters.

Monkeys practiced with a given decoder mapping (and subsequently updated versions of that mapping) for multiple days, defining a learning series. CLDA served two main purposes: (1) to enhance closed-loop performance from the initial decoder, and (2) to maintain performance despite shifts in neural activity, such as the loss of a unit within the BCI readout ensemble. The initial CLDA session (day 1) of a decoder series lasted approximately 5–15 minutes, providing the subject with sufficient performance to reach all targets successfully. Intermittent updates included (1) alterations to decoder parameters without changing the readout unit identities (weight change only) if there was a noticeable performance drop, assessed as a success rate decrease of approximately 10%–20%, and (2) interventions where readout unit identities were altered to address loss of a unit previously in the readout ensemble along with decoder weight changes (readout + weight change) (Fig. S1A). In these cases, CLDA was briefly run for 3–5 minutes, corresponding to 1-2 updates of the KF weights. Readout unit swaps and weight adjustments were made only if previously used readout units could no longer be isolated or identified. When a performance drop from the previous day was observed but all readout units remained present, only weight changes were made without altering readout units. CLDA therefore had two primary effects: initial CLDA led to large decoder changes that boosted performance to facilitate reaches to all targets, akin to initial "calibration"; mid-series CLDA made notably smaller adjustments to decoder parameters, which primarily maintained or modestly improved performance [28]

Monkey J and monkey S learned 13 and 6 adaptive decoder mappings in total, respectively [28]. Our analysis focused on extended learning series (longer than four days) that included intermittent CLDA on the KF decoder (monkey J: 7, monkey S: 3). Each learning series varied in length and continued until performance plateaued. Longer series tend to have more CLDA events (Fig. S1D). Information about the number of readouts and CLDA events in each learning series is shown in Supplementary table 1. Monkey S had a reduced frequency of mid-series CLDA, which does not necessarily imply better task performance than Monkey J, but may potentially reflect more stable neural recordings. Several of Monkey S's series were shorter than Monkey J's, which may have contributed to the reduced number of CLDA events. The original study intentionally manipulated the degree of initial CLDA to vary BCI performance on Day 1, with each learning series having different initial performances based on the extent of decoder adaptation (Fig. 1C, Fig. S1C). Some series began with lower initial performance due to minimal or no CLDA applied at the outset. Full details on adaptive decoder methods and the dataset are provided in Orsborn *et al.* [28].

Data analysis

Learning analyses and behavioral performance metrics

We quantified long-term learning changes during a series by comparing "early" and "late" training days. We defined an early training day as the first day with at least 25 trials per target direction. A late training day was the last day in the learning series with at least 25 trials per target direction.

We assessed behavioral improvements using standard performance metrics such as mean success percentage and reach times (Fig 1C). Success percentage was computed as the fraction of rewarded trials to the total number of trials initiated. Reach time was the time between cursor leaving center target and entering peripheral target.

Neural data preprocessing

Neural activity was binned at 100 ms and trial-aligned to the go-cue. We included neural activity from the onset of the go-cue to 1800 ms after as inputs to an offline decoding model (see below). Altering the time included (from go-cue to a range of 500 to 1800 ms post go-cue) was not found to qualitatively change reported findings (not shown). Because reach times vary over the course of learning, trials with a post-go-cue duration shorter than 1800 ms were zero-padded; this was most common in late phases of learning.

Logistic regression model

We analyzed changes in neural representations with multiclass logistic regression (LR) models that used neural activity (Figs. 2, 3) or transformations thereof (Fig. 5; see below), to predict target identity. The classifier model received flattened activity vectors composed of the activities of all units across time bins between go-cue to reach completion for each trial, and generated an output corresponding to the predicted target label. To control for variability in the number of trials completed to each target across days, we generated a training set where trials were randomly selected (without replacement) to match the number of trials to each target across days. We selected 25 trials per reach direction (200 trials per day), matching our criteria for defining "early" training phases. The remaining trials on a day were used as a test set. Unit firing rates were z-scored across trials for each set separately. We performed 100 training-test splits on each day, randomly selecting 200 trials for training (25 trials per reach direction) and designating the remaining trials for the test set in each draw. This approach allowed us to calculate the 95% confidence interval on classification accuracy (displayed as error bars). We used the mean accuracy across the training-test splits to compare classification accuracy across series and animals (Fig. 2E).

We implemented our LR models using the Python library Scikit-learn [59] with the L-BGFS solver and L2 regularization (hyperparameters: C = 1, max_iters = 1000). We did not observe any considerable changes in classification accuracy between LR models with or without regularization on late days (Fig. S2A). The hyperparameter C represents the inverse of regularization strength, with smaller values indicating stronger regularization, which helps prevent overfitting by penalizing large coefficients. We conducted a grid search for C ranging from C = 1e-5 to 1, across different neural populations (all, readouts, and nonreadouts). The purpose of this analysis was to confirm that our results were not merely due to the model performing unevenly across these groups. We found that classification accuracy was not more or less sensitive to the choice of C for any particular neural population (Fig. S2B). We note that this is not an analysis aiming to optimize the model's classification performance for each population.

Parameter values (model weights) for a single training-test split and for a single target (target 2) were used to illustrate the LR model (Fig. 2B). We quantified the similarity of decoder weights across days by calculating the similarity between the model weight matrices W on consecutive days, W^d, W^{d-1} (Fig. 2C), similar to previous analyses [16, 28]. This evaluation was performed by calculating the correlation coefficient between the flattened matrices of model weights (weights for each target concatenated) for each day.

Rank-ordered neuron adding curve

We estimated the contribution of each individual unit to target identity prediction by running a LR model for each unit separately (Fig. 3A, left). We then ranked units based on their classification accuracy (Fig. 3A, middle and Fig. 3B). A ranked-ordered Neuron Adding Curve (NAC) was produced by calculating the classification accuracy with the highest ranking unit, then with the first and second highest ranking units, and so on (Fig.3A, right and Fig.3C,F) [41]. These analyses used the same training-test split protocol described above. To account for the lower maximum classification accuracy on early days compared to later days, we normalized the NAC classification scores to each day's maximum performance using a linear scaling. We then computed the number of ranked units (N_c) required to achieve 80% normalized prediction accuracy for early and late days. We performed a Wilcoxon signed-rank test using the N_c's across series and monkey with alternative hypothesis $N_c^{\text{late}} < N_c^{\text{early}}$ (Fig. 3D,G).

The "combinatorial" NAC, as described in the main text, tested all possible combinations of units for each size of ensemble. We evaluated the separability between the classification accuracy distributions with and without the top N_c units (Fig. 3H-J) using a discriminability index, $d' = |m_1 - m_2|/[(s_1 + s_2)/2]$, where m_i and s_i are the mean and standard deviation of distribution i, respectively. A higher d' indicates that two distributions are more separable.

Classification using neural modes

We performed principal component analysis (PCA) on the readout activity for the early and late day separately (Fig. 5A). All trials with completed reaches were used for PCA each day, and readout firing rates were z-scored per day. We constructed PC adding curves and analyzed compaction in the PCs in the same way as in the above rank-ordered NAC, but using PCs instead single-unit activity (Fig. 5B-D). Importantly, we kept all PCs (no dimensionality reduction) and thus the number of PCs was equal to the number of readouts. We compared classification accuracy in low and high-variance PCs by splitting PCs into two groups (top 50% and bottom 50%) (Fig. 5F). These groups were defined by ranking PCs according to variance explained and assigning the first (last) $N/2$ PCs into the top (bottom) group.

PC loadings

The loadings are the components of the vectors produced by PCA, which generate the PCs by projecting a random vector variable onto them [60]. After normalizing the loading vector, we took the square of a readout's loading to represent its relative contribution to the PC. For each learning series, on late days, we identified the PC with the highest variance and the PC with the strongest classification accuracy (Fig. 5E), and the corresponding loading vector. For these PCs, we plotted the squared PC loading for each unit against its average rank based on its classification accuracy (Fig. 5H). We calculated the average rank for a unit from the distribution of their ranks across the training/test splits. Finally, we computed the sum of squared loadings (that we called the PC loading proportion) for the N_c leading and trailing units, according to their average rank (Fig. 5I).

Statistics

To evaluate statistical differences between early and late phases of learning, we applied the Wilcoxon signed-rank test for pairwise comparison. Significance levels are indicated as follows: ****: $p \leq 10^{-4}$, ***: $10^{-4} < p \leq 10^{-3}$, **: $10^{-3} < p \leq 10^{-2}$,*: $10^{-2} < p \leq 0.05$, ns: $p > 0.05$. The absence of a statistics comparison line in figures between two groups indicates that no statistically significant difference was found. Unless specified otherwise, the sample size for all such early vs late comparisons was $N = 10$ learning series for the BCI task, consisting of 3 series from monkey S and 7 series from monkey J. Inclusion of all learning series as opposed to only extended (> 4 days) series, did not qualitatively change the trends observed (not shown). In the supplementary material, control data from $N = 6$ learning series obtained from monkey J related to the arm movement task were used (Suppl. Figs. S2D,E and S3C).

Model

Simulations were implemented in C++ using the Eigen library [61]. Analyses were performed in Python, using the Numpy and Scipy packages. Values for parameters are included in Table 1. Code will be made publicly available upon publication.

Arm model

Our RNN model was initially trained to control a planar torque-based arm [62] consisting of a shoulder and an elbow joint. The arm only moved in the horizontal $x-y$ plane. The dynamical variables were the angle of each joint ($\mathbf{q} = [q_1, q_2]^\mathsf{T}$), their velocity ($\dot{\mathbf{q}}$) and the applied torque ($\boldsymbol{\tau}$). These were linked through the dynamics

$$\mathscr{M}(\mathbf{q})\ddot{\mathbf{q}} + \boldsymbol{c}(\mathbf{q}, \dot{\mathbf{q}}) + \mathscr{B}\dot{\mathbf{q}} = \boldsymbol{\tau} \tag{1}$$

where

$$\mathscr{M}(q_2) = \begin{bmatrix} a_1 + 2a_2 \cos q_2 & a_3 + a_2 \cos q_2 \\ a_3 + a_2 \cos q_2 & a_3 \end{bmatrix}$$

is a 2×2 inertia matrix,

$$\boldsymbol{c}(q_2, \dot{q}_1, \dot{q}_2) = a_2 \sin q_2 \begin{bmatrix} -\dot{q}_2(2\dot{q}_1 + \dot{q}_2) \\ \dot{q}_1{}^2 \end{bmatrix}$$

represents the centripetal and Coriolis force vector and

$$\mathscr{B} = \begin{bmatrix} 0.05 & 0.025 \\ 0.025 & 0.05 \end{bmatrix}$$

is the joint friction matrix. Parameters a_1, a_2 and a_3 are functions of the moments of inertia (I), masses (m), lengths (l) and distances from joint center to center of mass (d) of the two links (see Table 1), with $a_1 = I_1 + I_2 + m_2 l_1^2$, $a_2 = m_2 l_1 d_2$ and $a_3 = I_2$. The torques corresponded roughly to muscle activations and were related to the control signals $\mathbf{u}$ generated by the network dynamics by $\tau_f \dot{\boldsymbol{\tau}} = -\boldsymbol{\tau} + \mathbf{u}$ [63]. In simulations, these equations were discretized in time using the standard Euler scheme. The RNN defined below generated controls $\mathbf{u}_0, \ldots, \mathbf{u}_{T-1}$ and the arm's dynamics produced a sequence of states starting from an initial condition $[\mathbf{q}_0; \dot{\mathbf{q}}_0; \boldsymbol{\tau}_0]$. The position, velocity and acceleration of the end effector (hand) are denoted by $\mathbf{x}$, $\dot{\mathbf{x}}$ and $\ddot{\mathbf{x}}$, respectively. The initial condition was that the end effector be motionless ($\dot{\mathbf{x}}_0 = \ddot{\mathbf{x}}_0 = \mathbf{0}$) with its initial position $\mathbf{x}_0$ drawn from a uniform distribution centered on the center of the workspace, with radius 0.5 cm. The end effector position was obtained by a transformation of the joint coordinates, with the length of each link, l_1 and l_2, as parameters: $\mathbf{x} = [l_1 \cos q_1 + l_2 \cos(q_1 + q_2), l_1 \sin q_1 + l_2 \sin(q_1 + q_2)]^\mathsf{T}$. The corresponding end effector velocity was $\dot{\mathbf{x}} = J(\mathbf{q})\dot{\mathbf{q}}$, where $J(\mathbf{q})$ is the Jacobian of the transformation from $\mathbf{q}$ to $\mathbf{x}$. An expression for the end effector acceleration, needed in the learning objective described below, was obtained by differentiating $\dot{\mathbf{x}}$ with respect to time.

Decoder

The RNN model was then trained to perform BCI control. As in Ref. [28], the BCI decoder was a Kalman filter, using arm trajectories and neural activities to encode the Kalman model parameters, and then inverting the state and observation equations (defined below) to decode newly recorded neural activity. In discrete time, the state and observation equations read

$$\begin{aligned} \dot{\mathbf{x}}_t &= A\dot{\mathbf{x}}_{t-1} + \mathbf{w}_{t-1} \\ \mathbf{y}_t &= \mathbf{m} + C\dot{\mathbf{x}}_t + \mathbf{q}_t \end{aligned}$$

where $\mathbf{q}_t \sim \mathscr{N}(\mathbf{0}, Q)$ and $\mathbf{w}_t \sim \mathscr{N}(\mathbf{0}, W)$ are independent Gaussian random vectors, $\mathbf{y}_t$ is the firing activity and $\mathbf{m}$ is the average rate. The dimension of vectors $\mathbf{y}_t$, $\mathbf{m}$ and $\mathbf{q}_t$ was the same as the number of readouts in the model, $N_{\text{readout}} = 12$. Note that only the end effector velocity $\dot{\mathbf{x}}$ appears here, corresponding to a scenario where uncertainty on velocity does not propagate to position [23]. The position was determined by integrating the velocity.

Matrices A, W, C and Q were determined from simulation data during arm control following the algorithm described in Ref. [20]. To decode the cursor velocity while measuring neural activity online, we used the update equations

$$\begin{aligned} \hat{\dot{\mathbf{x}}}_t &= A\hat{\dot{\mathbf{x}}}_{t-1} + K_t(\mathbf{y}_t - \mathbf{m} - CA\hat{\dot{\mathbf{x}}}_{t-1}) \\ P_t &= (I - K_t C)(AP_{t-1}A^\mathsf{T} + W), \end{aligned}$$

where $K_t = P_t^- C^\mathsf{T}(CP_t^- C^\mathsf{T} + Q)^{-1}$ is the Kalman gain with $P_t^- = AP_{t-1}A^\mathsf{T} + W$. For simplicity, we set the initial estimate $\hat{\dot{\mathbf{x}}}_0 = \mathbf{0}$ with perfect precision, so that the error covariance matrix was $P_0 = 0$.

Closed-loop decoder adaptation (CLDA) was implemented following SmoothBatch [20, 28] using data recorded under brain control. Recorded velocities were rotated towards the current target, representing intended velocities [23]. This data with the rotated velocities produced new estimates for matrices C and Q and vector $\mathbf{m}$, denoted $\hat{C}$, $\hat{Q}$ and $\hat{\mathbf{m}}$, and the values from the last

Table 1. Parameters of the model. For the arm model, we combined data from Ref. [64] and from the morphology of the subjects in Ref. [28].

Parameters of the arm model			
symbol	*value*	*unit*	*description*
m_1	0.25	kg	mass of upper arm link
m_2	0.20	kg	mass of lower arm link
l_1	0.12	m	length of upper arm link
l_2	0.16	m	length of lower arm link
d_1	0.075	m	distance from center upper arm to center of mass
d_2	0.075	m	distance from center lower arm to center of mass
I_1	0.0018	kg·m^2	moment of inertia of upper arm
I_2	0.0012	kg·m^2	moment of inertia of lower arm
τ_f	40	ms	time constant connecting torques and controls
Parameters of the network dynamics			
N	100	—	number of units
α_v	0.2	—	reciprocal of time constant
σ	0.02	—	noise intensity
Parameters of the objective			
δ_p	0.01	m	scaling factor for position
δ_v	0.02	m/s	scaling factor for velocity
δ_a	0.08	m/s^2	scaling factor for acceleration
γ_v (arm)	0.25	—	velocity cost hyperparameter for arm control
γ_v (BCI)	0.1	—	velocity cost hyperparameter for BCI control
γ_a	0.05	—	acceleration cost hyperparameter for arm control
λ_u	0.005	N^{-2}·m^{-2}	effort penalty hyperparameter
Parameters of the learning algorithm			
α_R	0.3	—	relative weight factor for reward traces
—	2×10^{-5}	—	learning rate

step were replaced according to

$$X \leftarrow \alpha \hat{X} + (1-\alpha)X, \tag{2}$$

where $X \in \{C, Q, \mathbf{m}\}$ and $0 \leq \alpha \leq 1$ is the *CLDA intensity*; $\alpha = 0$ corresponds to a fixed decoder. In the model, CLDA was performed each day irrespective of performance, unlike in experiments, where within-series CLDA was applied only when performance declined.

Network model

The RNN contained N units fully connected by a recurrent weight matrix W (see Table 1). The units' output were their firing rates $\mathbf{r}$, which were obtained from their "membrane potential" $\mathbf{v}$ by applying an activation function $\boldsymbol{\varphi}(\cdot)$ elementwise: $\mathbf{r}_t = \boldsymbol{\varphi}(\mathbf{v}_t)$, i.e., $r_{t,i} = \varphi(v_{t,i})$ for each unit i. The transfer function φ was the rectified linear unit (ReLU), producing non-negative firing rates. The membrane potentials obeyed

$$\mathbf{v}_{t+1} = \mathbf{v}_t + \alpha_v[-\mathbf{v}_t + W\boldsymbol{\varphi}(\mathbf{v}_t) + \mathbf{i}_t + \mathbf{b}] + \sigma\sqrt{\alpha_v}\boldsymbol{\xi}_t \tag{3}$$

where $t = 0, 1, \ldots, T-1$. These dynamics integrate with time constant α_v^{-1} a recurrent input from other units in the network ($W\boldsymbol{\varphi}(\mathbf{v})$), an external input representing information about the task ($\mathbf{i}$)—e.g., the position of the target to reach; see below—, a bias ($\mathbf{b}$) and a zero-mean Gaussian white noise ($\sigma\sqrt{\alpha_v}\boldsymbol{\xi}$). The initial condition for the membrane potential was drawn from a uniform distribution $\mathscr{U}(-1,1)$.

The input $\mathbf{i}$ was given by $\mathbf{i}_t = U\mathbf{r}_t^{\text{in}}$, where U is the input weight matrix and $\mathbf{r}_t^{\text{in}}$ is the activity of the input layer. This input layer activity encoded premotor information. The encoding was a simple random projection of delayed information about target position $\mathbf{d}$, end effector position $\mathbf{x}$ and context $\mathbf{c}$—either $[1,0]^\mathsf{T}$ for arm control or $[0,1]^\mathsf{T}$ for BCI control—followed by the ReLU operation. Symbolically, $\mathbf{r}_t^{\text{in}} = \boldsymbol{\varphi}(F[\mathbf{x}_{t-l};\mathbf{d};\mathbf{c}])$, where F is a non-learnable random matrix and $l = 10$ is the delay. The output of the network—the controls applied to the arm models—was a linear mapping of the network activity [65]: $\mathbf{u}_t = V\mathbf{r}_t = V\boldsymbol{\varphi}(\mathbf{v}_t)$, where V is the output matrix.

Training objective

Let $\mathbf{d}_k$, $k = 0, \ldots, K-1$, represent the position of the $K = 8$ peripheral targets at a distance $D = 7$ cm from the center target, with $\mathbf{d}_k = D[\cos(2\pi k/K), \sin(2\pi k/K)]^\mathsf{T}$. The training objective was to minimize

$$L = \sum_{k=0}^{K-1} L_k \tag{4}$$

where

$$L_k = \frac{\|\mathbf{x}_T - \mathbf{d}_k\|^2}{\delta_p^2} + \gamma_v \frac{\|\dot{\mathbf{x}}_T\|^2}{\delta_v^2} + \gamma_a \frac{\|\ddot{\mathbf{x}}_T\|^2}{\delta_a^2} + \frac{\lambda_u}{T}\sum_{t=0}^{T-1}\|\mathbf{u}_t\|^2. \tag{5}$$

Therefore, the objective was to reach the target at the end of the trial (first term) with vanishing velocity (second term) and acceleration (third term), and with an effort penalty (fourth term). Parameters δ_p, δ_v and δ_a were used to rescale the position, velocity and acceleration terms. Hyperparameters γ_v, γ_a and λ_u controlled the relative weight of the velocity, acceleration and effort costs with respect to the position loss. Hyperparameters λ_u and γ_a were nonzero only under arm control (see Table 1).

Learning

The recurrent weights W, input weights U and biases $\mathbf{b}$ were plastic while the output matrix V and the encoding matrix F were fixed. Learning was performed *via* node perturbation using the REINFORCE algorithm [42, 66]. The noise $\boldsymbol{\xi}$ independently applied to all units in Eq. 3 evoked small end effector jitters. In reinforcement learning, jitters that increase reward should be reproduced, and network parameters should be updated accordingly. Here, the reward was taken to be minus the learning objective, $R = -L$. The gradient with respect to W was estimated using

$$\nabla_W R = \frac{\sqrt{\alpha}}{\sigma}\sum_{k=0}^{K-1}\left\{\left(R_k - \bar{R}_k\right)\sum_{t=0}^{T-1}\boldsymbol{\xi}_t^{(k)}\mathbf{r}_t^{(k)\mathsf{T}}\right\},$$

where $\boldsymbol{\xi}_t^{(k)}$ and $\mathbf{r}_t^{(k)\mathsf{T}}$ are the noise and activity when target k is presented. The reward trace $\bar{R}_k$ provided an estimate of the expected reward for target k [67] and was computed as a moving average

$$\bar{R}_k = \alpha_R \bar{R}_k^{\text{prev}} + (1-\alpha_R)R_k, \tag{6}$$

where $\alpha_R \in [0,1]$ is a factor that weights the relative contribution of the present reward (R_k) and the expected reward in memory ($\bar{R}_k^{\text{prev}}$) in computing the new reward estimate. For U, one simply replace $\mathbf{r}_t$ by $\mathbf{r}_t^{\text{in}}$; for $\mathbf{b}$, the summation is over $\boldsymbol{\xi}_t$ only. Parameter updates were computed after seeing all K targets (an epoch) using Adam updates with standard hyperparameters [68].

Analysis of the model

To allow clear comparison across seeds (i.e., random initializations of the network's parameters) and CLDA intensities, we normalized the BCI training loss L relative to the loss with a fixed decoder, denoted $L^{(0)}$. For each seed, we computed the $\log_{10}$ of L and $L^{(0)}$ for the last epoch of each day and we defined a new performance metric ("normalized log performance", Fig. 5C) as

$$P_d = \frac{\log_{10}(L_1^{(0)}) - \log_{10}(L_d)}{\log_{10}(L_1^{(0)}) - \log_{10}(L_{-1}^{(0)})}, \tag{7}$$

where $L_1^{(0)}$ and $L_{-1}^{(0)}$ are the losses with a fixed decoder on the first and last day, respectively. This linear transformation sought to transform the loss for the fixed decoder to the interval $[0,1]$.

Single-unit normalized performances (Fig. 5E) were computed by first evaluating the reach performance for each individual unit and ranking them from the most important to the least important. The performance of each unit was measured by restricting the Kalman filter parameters $\mathbf{m}$, C and Q to that unit and recomputing the Kalman gain K_t accordingly. This ensured that only that unit was controlling the cursor. Now, let $L_1, \ldots, L_{N_{\text{readout}}}$ be the loss of the most dominant unit (L_1) to the least dominant unit ($L_{N_{\text{readout}}}$) for a given seed and for a given value of CLDA intensity. The single-unit normalized reach performance was defined by

$$S_i = \frac{L_{N_{\text{readout}}} - L_i}{L_{N_{\text{readout}}} - L_1}. \tag{8}$$

We thus have $S_1 = 1$ and $S_{N_{\text{readout}}} = 0$.

A similar transformation was performed to obtain the normalized performance in Fig. 5F. If L_i is the loss after the ith most dominant unit has been added to the pool of readout units, then the normalized performance was

$$\hat{R}_i = \frac{L_1 - L_i}{L_1 - L_{N_{\text{readout}}}}. \tag{9}$$

Here, we have $\hat{R}_1 = 0$ and $\hat{R}_{N_{\text{readout}}} = 1$.

Combinatorial unit-adding curves (Fig. 5G) were computed by sampling all combinations of size s, for $s = 1, \ldots, N_{\text{readout}}$, and evaluating the loss when each such combination was moving the cursor. To allow comparison across CLDA intensities, we normalized each loss using the minimum and maximum losses (across all subset sizes) when a fixed decoder was used:

$$\frac{L_{\text{min}}^{(\text{fixed})} - L_c}{L_{\text{min}}^{(\text{fixed})} - L_{\text{max}}^{(\text{fixed})}}, \tag{10}$$

where L_c is the loss incurred for a specific combination.

The coefficients of determination in Fig. 5H were computed using the function `linregress` from the Python package `scipy.stats`. Each point in the scatter plots represents the change in a recurrent weight during learning with an adaptive decoder versus the weight change of the same connection with a fixed decoder, for a given network realization (random seed). Each plot contains 100^2 weights for each of the 10 realizations.