Self-Healing Neural Codes

M. E. Rule, T. O'Leary

March 8, 2021

Abstract

Neural representations change, even in the absence of overt learning. To preserve stable behavior and memories, the brain must track these changes. Here, we explore homeostatic mechanisms that could allow neural populations to track drift in continuous representations without external error feedback. We build on existing models of Hebbian homeostasis, which have been shown to stabilize representations against synaptic turnover and allow discrete neuronal assemblies to track representational drift. We show that a downstream readout can use its own activity to detect and correct drift, and that such a self-healing code could be implemented by plausible synaptic rules. Population response normalization and recurrent dynamics could stabilize codes further. Our model reproduces aspects of drift observed in experiments, and posits neurally plausible mechanisms for long-term stable readouts from drifting population codes.

1 Introduction

The cellular and molecular components of the brain change over time. In addition to synaptic turnover [1–3], ongoing reconfiguration of the tuning properties of single neurons has been seen in hippocampus [4, 5] and neocortex, including parietal [6], frontal [7], prefrontal [8], visual [9, 10], and olfactory [11] cortices. Remarkably, the reconfiguration observed in these studies occurs in the absence of any obvious change in behavior, task performance, or perception. How can we reconcile this stability with widespread ongoing changes in how the brain encodes experiences?

These recent and widespread observations seem to be at odds with well established evidence of homeostasis in neural circuit properties. Homeostasis is a feature of all biological systems, and examples of homeostatic plasticity in the nervous system are pervasive (e.g. [12–14]; and [15] for review). Broadly speaking, homeostatic plasticity is a negative feedback process that maintains physiological properties such as average firing rates, and distributions of synaptic strengths. This results in maintenance of collective properties, such as the total synaptic drive to a neuron or an average firing rate in a population. Regulation of collective properties is consistent with substantial variability in internal components ([16, 17]). This suggests that known homeostatic mechanisms may be capable of maintaining a consistent readout from a continually reconfiguring code [18, 19].

In this paper, we show that two kinds of homeostatic plasticity can stabilize a population code despite drift. We first argue that single-cell processes can stabilize the information-coding capacity of populations. We then describe a novel form of homeostatic plasticity that allows consolidated representations to interoperate with unstable neural populations. The implication of this finding is that long term storage of memories and percepts is possible dynamically, with relatively simple, known mechanisms. This potentially reconciles stable behavior with representational drift, as well as suggesting a mechanism that allows continual learning of multiple tasks without catastrophic interference.

Classical theories view homeostasis as processes within neurons that stabilize electrophysiological properties ([16, 17]), firing rates (e.g. [20–27]), variability [28–30], or synaptic weights (e.g. [31–35]). Importantly, homeostasis counteracts the destabilizing effects of Hebbian plasticity on both single-neuron [36–39] and network activity statistics [40–44]. Recent studies predict that single-cell homeostasis should also stabilize second-order statistics, like pairwise correlations [44]. In our model of representational drift, these processes maintain selectivity and function in neural population codes, while allowing individual neurons to reconfigure.

We also develop a second sense of homeostasis that allows consolidated representations to maintain stable relationships with unstable neural population codes. This form of homeostasis arises from the interaction between single-cell homeostatic processes, and Hebbian learning in a predictive coding framework [45–49]. When combined with recurrent network dynamics, such "Hebbian homeostasis" proves to be a powerful tool for stabilizing consolidated neural representations in the presence of drift.

The mechanisms we propose here are theoretical, but they are grounded in well-established principles of neuronal function. Our model therefore yields testable predictions about how Hebbian plasticity and homeostasis should interact to stabilize neural representations.

1.1 Background

We briefly review representational drift and the broader context of the ideas used in this manuscript. Representational drift refers to seemingly random changes in neural responses during a learned task that are not associated with learning [18]. Potential causes include ongoing learning of unrelated tasks [50], noisy fluctuations (e.g. [2]), and time-stamping activity for episodic memory [51].

For example, in Driscoll et al. [6] mice navigated to one of two endpoints in a T-shaped maze (Figure 1a), based on a visual cue. Population activity in Posterior Parietal Cortex (PPC) was recorded over several weeks using fluorescence calcium imaging. Neurons in PPC were tuned to the animal's past, current, and planned behavior. Gradually, the tuning of individual cells changed: neurons that were initially selective to, e.g., the beginning of the maze, could start to fire more toward the end—or become disengaged from the task entirely (Figure 1b). The neural population code eventually reconfigured completely (Figure 1c). However, neural tunings continued to tile the task, indicating stable task information at the population level. These features of drift have been observed throughout the brain [5, 9, 10].

Gradual drift would be relatively easy for a downstream readout to track using external error feedback , e.g. from ongoing rehearsal [19]. Indeed, recent simulation studies confirm that learning in the presence of noise can lead to a steady state, in which drift is balanced by error feedback [52, 53]. Here, we will show that it is possible to track drift without an external learning objective.

Previous studies have shown how stable functional connectivity can be maintained despite synaptic turnover [42, 54, 55]. However, we are precisely interested in the scenario where functional connectivity itself is unstable, allowing the roles of single neurons to change. Additionally, recent work has shown that discrete representations can be stabilized despite drift using neural assemblies [56–59]. Self-correcting assemblies provide a compelling model for the longevity of discrete information, such as semantic knowledge. However, we argue that none of these models fully explain how the brain can maintain stable sensorimotor representations despite drift.

Neural assemblies are populations of cells that can exhibit self-excitatory, self-sustaining activity. Once learned [60], assemblies can be maintained without external training [58, 59, 61]. Since assembly activation is all-or-nothing, no fidelity is lost if a few neurons enter or leave the assembly. A readout can detect this, and update how it interprets neural population activity [59]. This mechanism is essentially binary: it combines a majority-sum error correcting code with plasticity to adapt readouts to changing neural tuning.

However, the brain must contend with continuous sensorimotor variables. Recent experiments suggest that neural representations of these variables are also continuous [62]. Even if internal representations are discrete [63–65], the external world is not. Some states will always lie at ambiguous boundaries between different assemblies. Here, small amounts of drift can introduce large changes.

Despite this, neural representations of continuous tasks are stable. Neural activity is typically confined to a low-dimensional manifold that reflects sensory, motor, and cognitive variables (e.g. [66–69, 69–82]). The geometry of these low-dimensional representations is consistent over time, although the way it is reflected in neuronal firing changes [81, 83]. Engineers have applied online recalibration and transfer learning and to track drift in brain-machine interface decoders ([84–87]; see [88] for review). Could neurons in the brain do something similar? We argue that neuronal homeostasis and Hebbian plasticity driven by internally-generated prediction errors allows neural networks to, in effect, “self-heal".

2 Results

Here, we explore how neural networks could track drift in sensorimotor representations. There are two important general principles to keep in mind throughout. First, distributed neural representations are massively redundant. To create ambiguity at the macroscopic level, many smaller disruptive changes must occur in a coordinated way. Neurons can exploit this to improve their robustness to drift. Second, learning creates recurrent connections that allow neural populations to predict their own inputs and activity. Even if learning has ceased, these connections remain. This allows a downstream readout to repair inputs corrupted by drift, and use these error-corrected readouts as a training signal.

In the first half of the manuscript, we discuss how homeostasis achieves stable population-level representations, despite instability in single-neuron tunings. We then explore how a single neuron might stabilize its own readout in the presence of drift using homeostasis, and updating its synaptic weights. In the latter half of the manuscript, we show that these rules imply a form of Hebbian learning that achieves homeostasis. We extend these ideas to neural populations, and show that recurrent dynamics can stabilize a readout of an unstable neural code.

2.1 A model for representational drift

To understand how neurons might cope with unstable population codes, we must first build a model of representational drift. We focus on continuous representations of task space, like those studied in Ziv et al. [4] and Driscoll et al. [6], and simplify our model as much as possible.

Figure 1b illustrates average neuronal fluorescence intensities as a function of progress through the task, mapped to a pseudo-location variable $\theta \in [0, 1]$ (Methods: *Data and analysis*). Neurons fired preferentially in specific parts of the maze. Preferred tunings were typically stable, but occasionally changed abruptly. Figure 1c shows a population of forty neurons tracked over thirty-nine days. Neurons could be sorted according to their preferred location on the first day, and tiled the task space. Preferred tunings gradually switched over time to new locations, leaving little trace of the original code after a month.

To model this, we consider a population of N neurons "$\mathcal{X}$" that encodes states θ. We assume that the encoded states θ lie on a continuous low-dimensional manifold. We neglect noise, and assume that θ is encoded in the vector of instantaneous firing rates in a neural population, with tuning curves $\mathbf{x}(\theta) = \{x_1(\theta), .., x_N(\theta)\}^\top$.

The population statistics [4, 6, 11], and low-dimensional geometry [81, 83] of drifting population codes remains stable. The properties of single-neuron tuning curves are also preserved: place cells may change their preferred location, but always look like place cells [4]. We incorporate these constraints by viewing tuning curves as random samples from the space of possible tuning curves, constrained by the statistics of the encoded variables.

To define this random process, we assume that a task is associated with a set of K features, $\mathbf{s}(\theta) = \{s_1(\theta), .., s_K(\theta)\}^\top$. These features have a fixed relationship to the external world, for example visual input or the space of joint configurations,

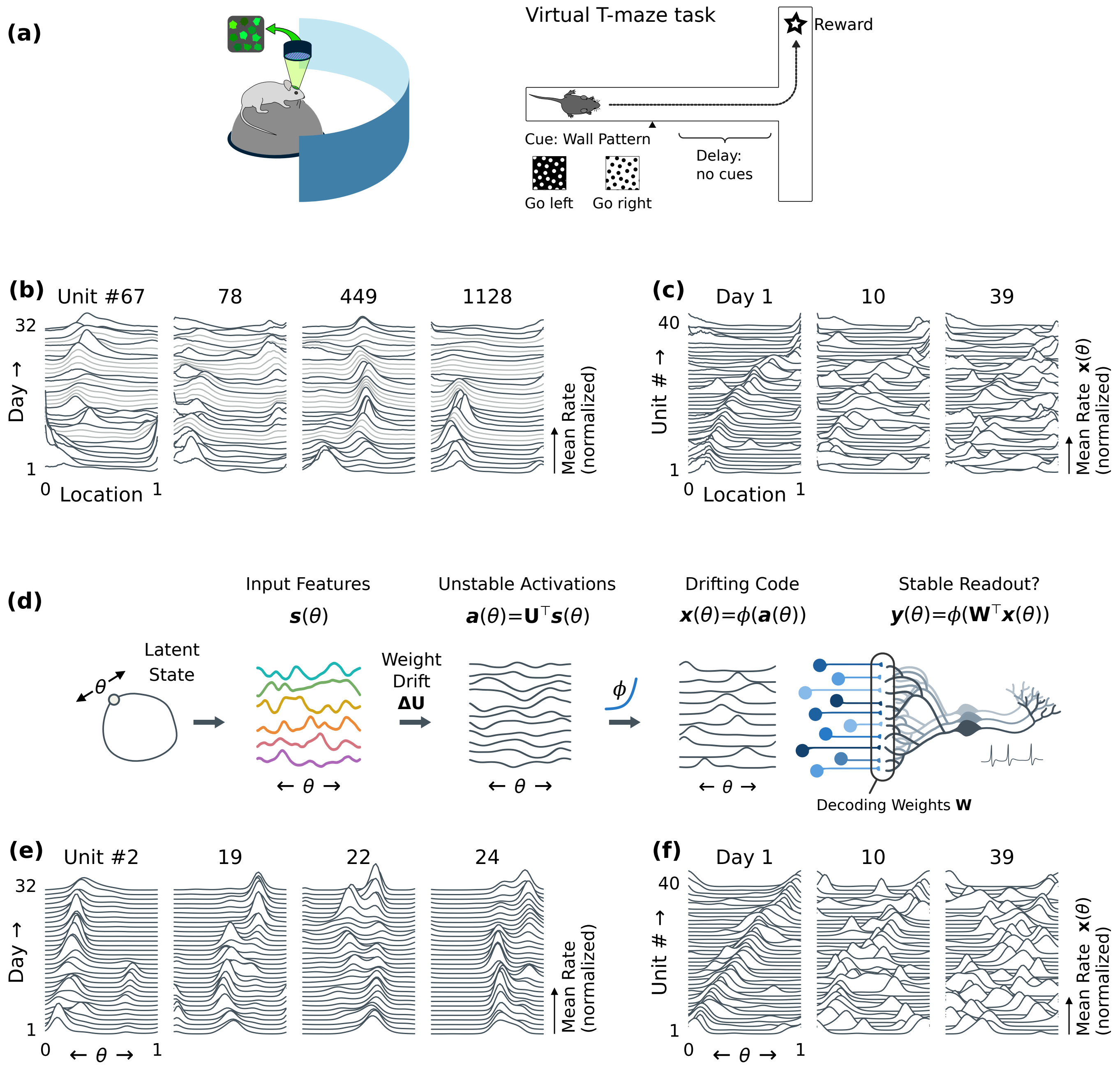

Figure 1: **A model for representational drift.** **(a)** Driscoll et al. [6] imaged population activity in PPC for several weeks, after mice had learned to navigate a virtual T-maze. Neuronal responses continued to change even without overt learning. **(b)** Tunings were often similar between days, but could change unexpectedly. Plots show average firing rates as a function of task pseudotime (0=beginning, 1=complete) for select cells from Driscoll et al. [6]. Tuning curves from subsequent days are stacked vertically, from day 1 up to day 32. Missing days (light gray) are interpolated. Peaks indicate that a cell fired preferentially at a specific location (Methods: *Data and analysis*). **(c)** Neuronal tunings tiled the task. Within a day, one can decode the mouse's behavior from population activity [6, 19]. Plots show normalized tuning curves for 40 random cells, stacked vertically. Cells are sorted by their preferred location on day 1. By day 10, many cells have changed tuning. Day 39 shows little trace of the original code. **(d)** We model drift in a simulated rate network (§2.1; Methods: *Simulated drift*). An encoding population $\mathbf{x}(\theta)$ receives input $\mathbf{s}(\theta)$ with low-dimensional structure, in this case a circular track with location θ. The encoding weights $\mathbf{U}$ driving the activations $\mathbf{a}(\theta)$ of this population drift, leading to unstable tuning. Homeostasis preserves bump-like tuning curves. **(e)** As in the data (a-c), this model shows stable tuning punctuated by large changes. **(f)** The neural code reorganizes, while continuing to tile the task. We will examine strategies that a downstream readout $\mathcal{Y}$ could use to update how it decodes $\mathbf{x}(\theta)$ to keep its own representation $\mathbf{y}(\theta)$ stable. This readout is also modeled as linear-nonlinear rate neurons, with decoding weights $\mathbf{W}$.

and capture the statistics of the encoded variables θ. To model this, we take $s(\theta)$ to be fixed samples from a Gaussian process on θ:

$$s(\theta) \sim \mathcal{GP}[0, \Sigma(\theta, \theta')] \tag{1}$$

These features are combined linearly through an encoding weight matrix $\mathbf{U} = [\mathbf{u}_1, .., \mathbf{u}_N]$, to yield the synaptic activations $\mathbf{a}(\theta)$=$\{a_1(\theta), .., a_N(\theta)\}^\top$ of the encoding population. Each column $\mathbf{u}_i$ is the encoding weights for a single unit x_i. The firing rates $\mathbf{x}(\theta)$ are then given as a nonlinear function of these activation functions:

$$\begin{aligned}\mathbf{a}(\theta) &= \mathbf{U}^\top \mathbf{s}(\theta)\\ \mathbf{x}(\theta) &= \phi[\mathbf{a}(\theta)]\end{aligned} \tag{2}$$

The nonlinearity $\phi[\cdot]$ can be any function that is rectifying and monotonically increasing, although we use the exponential here.

If the encoding weights are taken as i.i.d. samples from a standard normal distribution, $\mathbf{u}\sim\mathcal{N}(0, I_N)$, then the activation functions will follow a zero-mean Gaussian process on θ with covariance inherited from $\mathbf{s}(\theta)$. This converts the problem of defining drift as a random walk through the space of possible activation curves $\mathbf{a}(\theta)$, to a simpler random walk in the space of encoding weights, $\mathbf{U}$. (See Methods: *Simulated drift* for details of how these weights evolve, and why this preserves information about θ in the population.)

At this point we should pause to address two caveats of this model. First, the fixed features $\mathbf{s}(\theta)$ do not exist in a literal sense. It is true that primary sensory and motor connections are fixed, but these do not provide a sufficiently rich basis to describe all possible sensorimotor transformations. Richer representations are constructed through transformations within the brain (e.g. [89, 90]). The synapses involved in these transformations are also subject to drift. The decomposition we describe here, of fixed $\mathbf{s}(\theta)$, and drifting $\mathbf{a}(\theta)$, captures the abstract principles that (I) the brain has learned a rich representation of θ with fixed statistics, (II) this representation is tethered to the external world, and (III) drifting synaptic weights cause neurons to wander through the space of task-relevant tuning curves .

The second caveat we should address is that this model is not, on its own, especially stable. We have assumed that inputs $\mathbf{s}(\theta)$ follow a fixed distribution, that the encoding weights $\mathbf{U}$ follow a particular distribution, and that these coincidentally lead to just the right synaptic activations $\mathbf{a}(\theta)$ to yield sensible firing rates when passed through nonlinearity $\phi[\cdot]$. These constraints are easily enforced in a computer, but biological systems must achieve them through homeostatic tuning or regulation of the network activity.

To model these homeostatic processes, we impose an additional constraint on the mean and the variance of the firing rate for each encoding neuron $x_n(\theta)$:

$$\begin{aligned}\langle \mathbf{x}_n \rangle &= \mu_0\\ \mathrm{var}[\mathbf{x}_n] &= \sigma_0^2\end{aligned} \tag{3}$$

These moments are fixed by homeostatically adapting a bias β and gain γ of each neuron separately:

$$x(\theta) = \phi[\gamma\mathbf{a}(\theta) + \beta]. \tag{4}$$

The bias can be viewed as threshold adaptation, and the gain can be interpreted as synaptic scaling. These processes control the excitability and variability of the encoding neuron, respectively. They occur over hours to days, through homeostatic regulation in single neurons [28, 29]. For a fixed average firing rate, larger variability invariably corresponds to higher selectivity. Homeostatic regulation of these statistics ensures that (I) encoding neurons retain a reasonable range of firing rates and (II) the tuning curves of these encoding neurons remain selective for a particular preferred stimulus θ_0 (or a set of preferred stimuli that are similar in some way).

For encoding neuron $x(\theta)$, we adjust the gain and bias based on the error between the neuron's firing rate statistics, and the homeostatic targets (3).

$$\begin{aligned}\Delta\gamma &\propto \varepsilon_\sigma = (\sigma_0^2 - \mathrm{var}[x])/\sigma_0^2\\ \Delta\beta &\propto \varepsilon_\mu = \mu_0 - \langle x \rangle\end{aligned} \tag{5}$$

In general, these multiple homeostatic processes acting in parallel interact, potentially leading to instability [28, 29, 40, 91]. One solution is to allow threshold adaptation to be much faster than synaptic scaling. Another is for the synaptic scaling process to also adapt the threshold, canceling out any influence on excitability.

Figure 1 illustrates examples of tuning curve drift from Driscoll et al. [6], compared to the Gaussian-process model of drift described above. Figure 1d-f illustrates simulated tuning curve drift in the model. We define a circular environment, with a periodic location coordinate $\theta \in [0, 2\pi)$. This location drives fixed input features $\mathbf{s}(\theta)$, which then drive activity in the encoding population $\mathbf{x}(\theta)$ via encoding weights $\mathbf{U}$. Drift is simulated as a random walk on these encoding weights, and the encoding cells' tuning curves are homeostatically maintained according to (3) and (4). Further details can be found in the Methods: *Simulated drift*. Notably, the model mimics changes in tuning curves seen in Driscoll et al. (2017). In Figure 1e, we see that individual encoding neurons show a punctuated stability in their tuning, similar to Figure 1b. Likewise, Figure 1f shows that the tuning curves of the encoding population tile the state space, but gradually reconfigure over several weeks, similarly to Figure 1c.

Overall, this illustrates that neural population codes displaying drift similar to that seen in the brain arise under very generic circumstances. The only constraints are (I) that inputs to the population reflect the similarity space of the encoded variables θ, and (II) that some sort of homeostasis regulates neuronal excitability and selectivity on long timescales. In a larger population, single-cell homeostasis is sufficient for maintaining localized, bump-like tuning curves that tile the task (Figure 1f). Other processes, such as response normalization [92], could also provide this, and would be useful for ensuring an even tiling of the space in smaller populations.

2.2 Hebbian homeostasis stabilizes readouts without error feedback

Neural population codes are massively redundant. For example, most of the neural variability in Driscoll et al. [6] is explained by progress through the maze, conditioned on the current and planned turn direction. Nonlinear dimensionality reduction algorithms recover the latent T-shaped structure of the

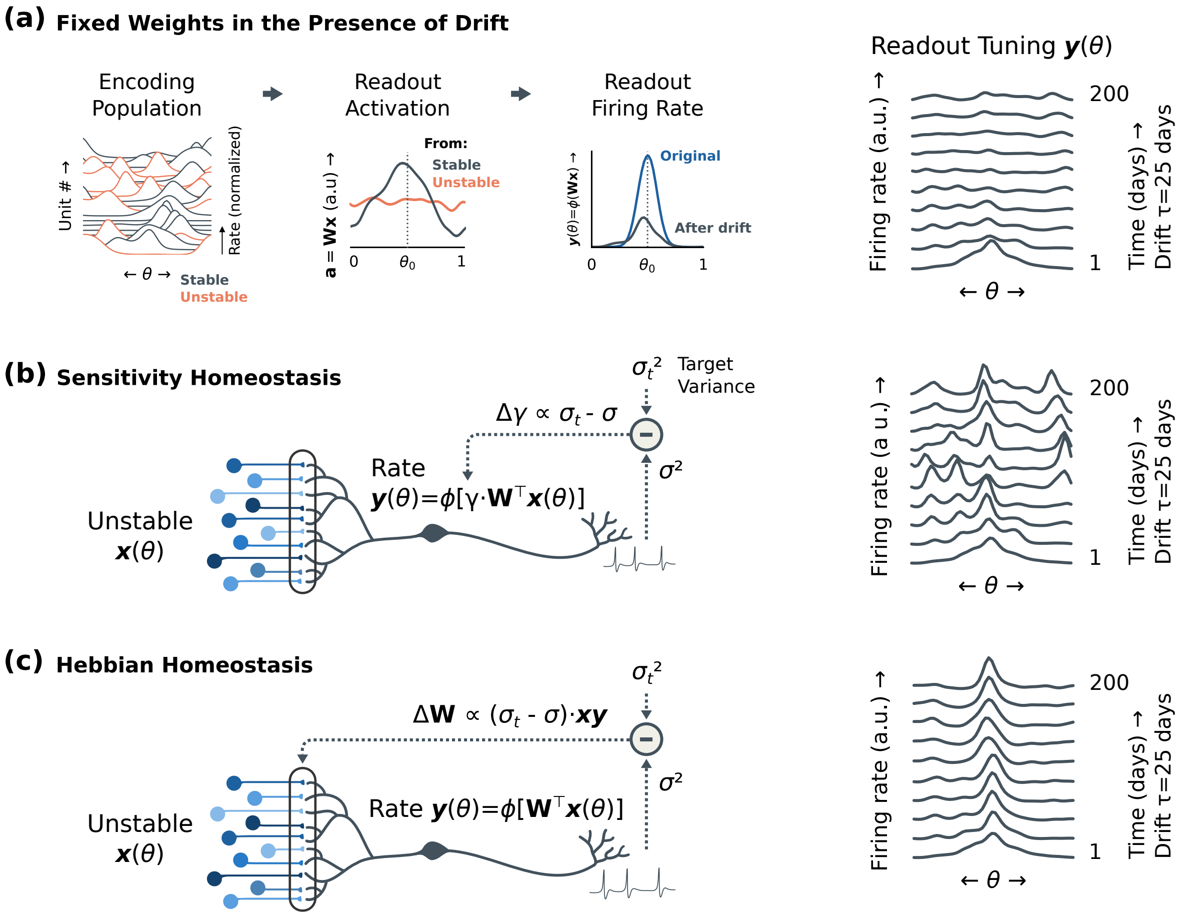

Figure 2: **Homeostatic Hebbian plasticity enables stable readout from unstable populations.** **(a)** Simulated linear-nonlinear units that are driven by redundant population activity show a loss of excitability, not a change in tuning, when their inputs drift. Since the cell is selective to a conjunction of features, it loses excitatory drive when some of its inputs change. Since most drift is orthogonal to this readout, however, the preferred tuning θ_0 does not change. The right-most plot shows that the excitability gradually diminishes as a larger fraction of inputs change. **(b)** Homeostatic adjustments to neuron sensitivity stabilizes readouts for small amounts of drift. As more inputs reconfigure, the cell compensates for loss of excitatory drive by increasing an effective gain parameter γ. However, the readout changes to a new, random location once a substantial fraction of inputs have reconfigured (right). This phenomenon is the same as the model for tuning curve drift in the encoding population (c.f. Fig. 1e). **(c)** Hebbian homeostasis increases neuronal variability by potentiating synaptic inputs that are correlated with post-synaptic activity, or depressing those same synapses when neuronal variability is too high. This results in the neuron relearning how to decode its own tuning curve from the shifting population code, supporting a stable readout despite complete reconfiguration (right).

task [18]. Because of redundancy, there are many valid ways to decode information from the population. We propose that, in the absence of external error feedback or sensorimotor rehearsal, a readout could use this to generate a surrogate error signal. The error signal supports a plasticity rule that could allow unstable neural codes to be continuously reconsolidated.

This self training re-encodes a learned readout function $\mathbf{y}(\theta)$ in terms of the new neural code $\mathbf{x}(\theta)$, allowing the network to track an unstable representation. Surprisingly, this "self-healing" plasticity stabilizes the readout of unstable population codes even in single neurons. We first sketch an example of this plasticity, and then explore why this works.

Using our drifting population code as input, we model a readout population of M neurons with tuning curves $\mathbf{y}(\theta) = \{y_1(\theta), .., y_M(\theta)\}^\top$ (Figure 1d). If this readout is stable, then the responses $\mathbf{y}(\theta)$ should remain fixed, even as the encoding population $\mathbf{x}(\theta)$ reconfigures completely. We model this decoder as a linear-nonlinear function, using decoding weights $\mathbf{W}$ and biases (thresholds) $\mathbf{b}$:

$$\mathbf{y}(\theta) = \phi[\mathbf{W}^\top \mathbf{x}(\theta) + \mathbf{b}]. \tag{6}$$

On each simulated day, we re-train the decoding weights using a Hebbian rule. This potentiates decoding weights whose input $\mathbf{x}_n(\theta)$ is correlated with the post-synaptic firing rate $y_m(\theta)$. We also adapt the threshold, $\mathbf{b}$, to maintain the average firing rate, and include some weight decay:

$$\begin{aligned} \Delta\mathbf{W} &\propto \varepsilon_\sigma[\langle \mathbf{x}(\theta)\mathbf{y}(\theta)^\top \rangle_\theta - \mathbf{W}] \\ \Delta\mathbf{b} &\propto \varepsilon_\mu \langle \mathbf{x}(\theta) \rangle_\theta . \end{aligned} \tag{7}$$

In some ways, (7) resembles the homeostatic rules explored earlier. Firing rate statistics are controlled through negative feedback, driven by measurements of the deviations from the target set-points ε_μ and ε_σ (3). However, rather than scale all weights uniformly, this rule adjusts the component of the weights that is most correlated with the postsynaptic output, $y(\theta)$.

Traditionally, “homeostatic Hebbian plasticity" refers to processes that stabilize synaptic weights and responses under ongoing rehearsal and learning [36–44, 54, 55]. The role of “Hebbian homeostasis" here is more specific: the neurons adjust their activity toward homeostatic set-points using Hebbian (or anti-Hebbian) learning.

Figure 2 illustrates the mechanisms and consequences of Hebbian homeostasis (7). It simulates a single neuron driven by the unstable population code. With fixed weights (Figure 2a), drift reduces the excitability without changing its tuning. This is because the readout requires a conjunction of specific inputs to fire. Drift gradually destroys this conjunction, and is unlikely to spontaneously create a similar conjunction at a different part of the coding space. A similar phenomena may underlie forms of drift that consist of changes in excitability, but stable preferred tuning [7, 9, 93]. For small amounts of drift, firing-rate homeostasis (5) can temporarily stabilize the readout (Figure 2b). Eventually, however, the encoding population $\mathbf{x}(\theta)$ reconfigures so drastically that no trace of the original code remains, and the cell acquires a new preferred stimulus.

In contrast, Figure 2c illustrates the consequences of Hebbian homeostasis. As the encoding population $\mathbf{x}(\theta)$ drifts, the overall excitatory drive to the neuron decreases. This activates homeostatic plasticity to restore neuronal excitability. However, instead of scaling up all synapses uniformly, the neuron selectively potentiates the component of $\mathbf{x}(\theta)$ that correlates with its own output. This leverages the fact that small amounts of drift change neuronal excitability, but not tuning. The neuron's own output provides a teaching signal to re-learn decoding weights for inputs that have changed.

If Hebbian homeostasis is applied continuously, a readout can track drift despite complete reconfiguration in the encoding population $\mathbf{x}(\theta)$. In effect, the readout's initial tuning curve is transported to a new set of weights that estimate the same function from an entirely different input. This homeostatic rule might seem like ad-hoc speculation. However, we will show that such a rule arises naturally in networks that learn through predictive coding, and is a plausible consequence of the interaction between prevailing models of learning and homeostasis.

2.3 Predictive models track drift

Neural populations learn internal models that recapitulate the statistics and dynamics of the external world [94, 95]. These internal models give rise to neural population codes that attempt to predict their own inputs [45, 46, 48, 49, 78]. Learning procedures based on minimizing prediction error lead to efficient spiking codes [96–98] that can perform statistical and dynamical computations [99, 100].

Predictive coding might also be how the brain maps behavioral errors to specific synaptic weight updates [47]. In essence, neurons predict the expected outcome of an action, and update their synaptic weights based on any error in this prediction. The error on a task is minimized only when the prediction errors between all brain areas involved in said task are also minimized. As a result, the overall cost function sampled during learning is gradually consolidated into a distributed representation throughout the brain, in the form of these local predictive models.

We propose that these internal models provide the error signals needed to integrate stable and volatile neural representations. There are two way to view this process. In one view, predictive models constrain neural activity, which can be used to detect and correct drift. In another view, the brain generates a teaching signal that trains neurons how to re-interpret the meaning of neurons whose function have changed, akin to the student-teacher framework in machine learning [101]. By computing this teaching from local recurrent dynamics, the brain continually re-trains itself, akin self-distillation [102].

We propose that a general strategy for tracking drift in a neural population should contain three components.

I The readout should leverage redundancy to minimize the error caused by drift.

II The readout should use its own activity as a training signal to update its decoding weights.

III The correlation structure of the readout population should be homeostatically preserved.

To show how these principles imply Hebbian homeostasis, we unpack them in a linear network. We then illustrate that

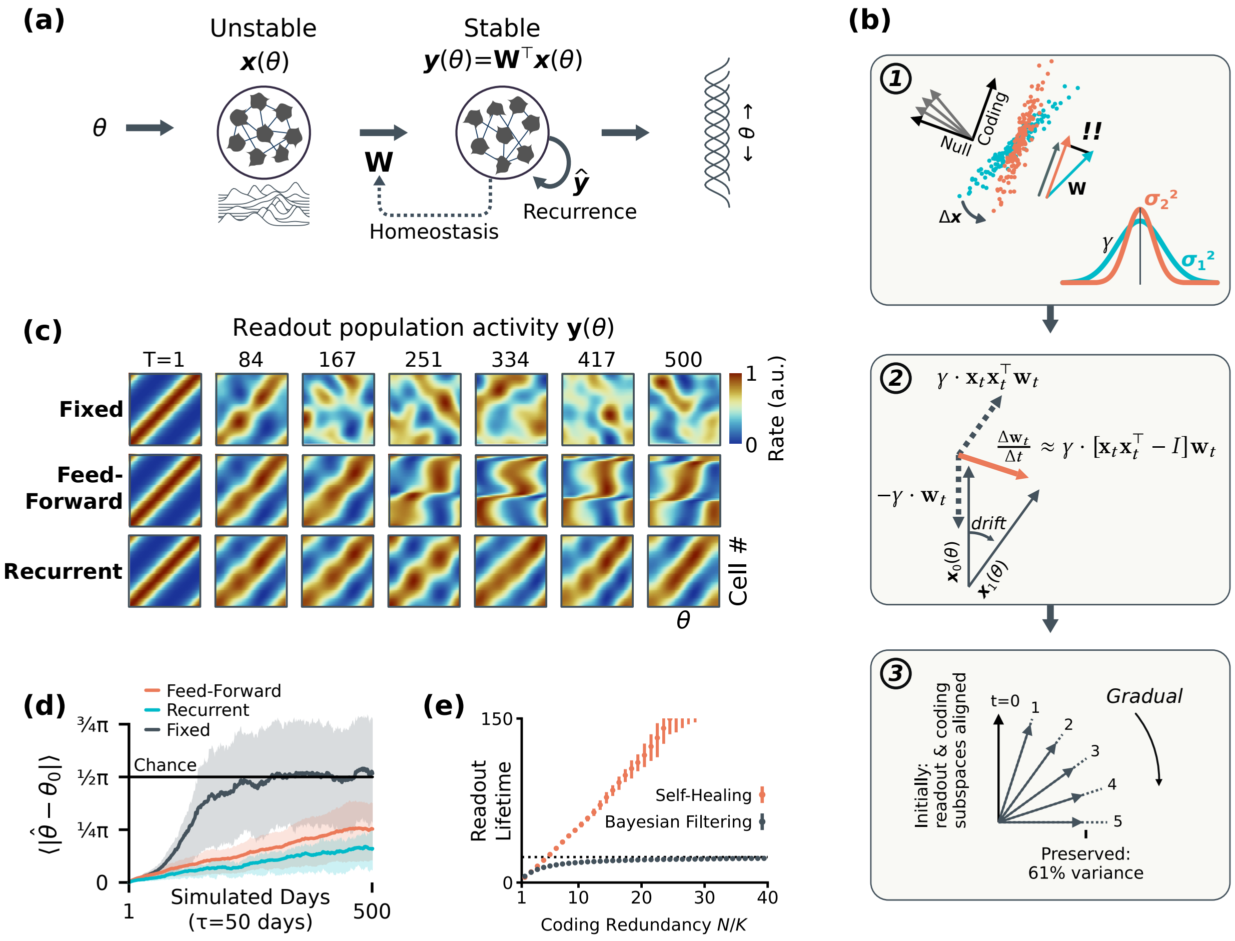

Figure 3: **Self-healing codes in a linear model.** **(a)** Network schema: An unstable population $\mathbf{x}(\theta)$ encodes variables θ (c.f. Fig. 1). A linear readout $\mathbf{y}(\theta)$ seeks to preserve its representation via Hebbian homeostasis, and can use recurrent activity as a training signal $\hat{\mathbf{y}}$. **(b-1)** Drift changes the embedded low-dimensional structure in population activity. Most drift occurs in non-coding directions, and readouts can detect when low-dimensional activity no longer aligns with their synaptic weights. In linear models, this corresponds to reduced firing-rate variability. **(b-2)** Hebbian homeostasis restores a target variability by re-aligning the decoding weights with low-dimensional activity. This is the sum of a Hebbian and weight-decay term, scaled by the homeostatic error γ. **(b-3)** For small amounts of drift, this self-repair has low (but nonzero) error. Large amounts of drift can be tracked if changes are gradual. **(c)** Stability of a readout $\mathbf{y}(\theta)$ with bump-like tuning curves tiling a circular space. Encoding cells $\mathbf{x}(\theta)$ drift with time-constant $\tau = 50$ days ("one epoch"). We simulate ten epochs, applying continuous-time Hebbian homeostatic learning rules (Eq. 13). Fixed weights degrade rapidly. Single-cell homeostasis provides some stability for ≈ 3 epochs, but preferred directions shift. Recurrent dynamics better preserve population correlation structure. **(d)** Hebbian homeostasis reduces the drift of the readout, and recurrence stabilizes it further. The ability of the linear network to error-correct is limited, so the readout still drifts in the long-term (but see Fig. 4). Shaded regions reflect the interquartile range over twenty random realizations. **(e)** Readout longevity scales linearly with the population code redundancy (red). Bayesian filtering alone decays rapidly in comparison (black; Eqs. 22-25, Methods: *Weight filtering in a linear model*). Here, we defined a K-dimensional neural code embedded randomly in an $N = 200$ dimensional population, with encoding weights undergoing random drift with a time constant $\tau = 20$. The readout lifetime was defined as the number of simulated days required for the readout Normalized Mean-Squared Error (NMSE; 1=chance) to reach 0.5. Error bars reflect the interquartile range for ten random network instances with five random realizations of drift for each network. (Methods: *Linear simulations*).

these principles lead to long-term stability, despite drift, in a nonlinear network.

2.4 A self-healing linear readout

In a linear network (Fig. 3a), the readouts $\mathbf{y}(\theta)$ can be viewed as the output of ordinary least-squares linear regression. Although this network is not particularly good at correcting errors, it does provide useful intuition.

We incorporate the three components of self-healing codes (robustness, self-training, and correlation homeostasis) as follows: (I) We regularize decoding weights to improve robustness; (II) We use the readout's own activity as a training signal; (III) We use firing-rate-variance homeostasis and recurrent dynamics to stabilize correlations.

We assume that the readout is initially trained from an external error signal, and consider a drifting population code $\mathbf{x}_d(\theta)$ that evolves randomly over several days "d". Given a training signal $\mathbf{y}_0$, the regularized least-squares solution for the ideal decoding weights for the following day $d+1$ is:

$$\mathbf{W}_{d+1} = [\Sigma_d + \Sigma_\Delta]^{-1} \langle \mathbf{x}_d \mathbf{y}_0^\top \rangle, \tag{8}$$

where $\Sigma_d = \langle \mathbf{x}_d {\mathbf{x}_d}^\top \rangle$ is the covariance of the encoding population on day d, and Σ_Δ is a regularizing term reflecting the expected covariance of day-to-day drift.

To incorporate self-training, we generate the training signal for the weights on day $d+1$ from the network's own output on day d. For a linear readout, the decoded $\hat{\mathbf{y}}$ is the linear projection $\mathbf{W}_d^\top \mathbf{x}_d$:

$$\mathbf{W}_{d+1} = [\Sigma_d + \Sigma_\Delta]^{-1} \Sigma_d \mathbf{W}_d \tag{9}$$

This update applies recursive filtering to the weights (see Methods: *Weight filtering in a linear model*). However, filtering alone is unhelpful (Fig. 3e), since it allows activity to decay as predictions become uncertain. To stabilize the firing-rate variability, we rescale the training signal to compensate for any loss of variability σ_y^2, from its homeostatic target σ_0^2. For a single readout neuron with weights $\mathbf{w}$, this gives the homeostatic update:

$$\mathbf{w}_{d+1} = [\Sigma_d + \Sigma_\Delta]^{-1} \Sigma_d \mathbf{w}_d \frac{\sigma_0}{\sigma_y} \tag{10}$$

This update can be solved by online stochastic gradient descent using a Hebbian learning rule (Methods: *Synaptic learning rules*).

$$\Delta \mathbf{w}_t \propto \gamma\, \mathbf{x}_t \mathbf{y}_t^\top - \Sigma_\Delta \mathbf{w}_t \tag{11}$$

We can use loss of excitatory drive as an indicator of the current drift rate, setting $\hat{\Sigma}_\Delta \approx \gamma I$ (Methods: *Estimating the rate of drift*). This gives the Hebbian rule:

$$\Delta \mathbf{w}_t \propto \gamma \cdot [\mathbf{x}_t \mathbf{x}_t^\top - I] \mathbf{w}_t \tag{12}$$

This learning rule is the same as the Hebbian homeostasis rule proposed earlier (7). Its acts as follows: In redundant, low-dimensional codes, most drift occurs in directions that are not used for coding (Fig. 3b-1). Drift does, however, reduce input drive to a readout. Neurons can detect this, and apply Hebbian homeostasis to re-align their decoding weights with the encoding subspace (Fig. 3b-2). This process allows synaptic weight to track drift as it occurs. If drift is gradual, a stable readout can survive multiple complete reconfigurations of the input code (Fig. 3b-3).

This update resembles classic, linear approximations to Hebbian learning [103, 104] with weight decay. Such learning rules extract the leading principle component(s) of their input, $\mathbf{x}$. Indeed, the self-healing weight update will cause the readout to regress towards the leading eigenmode(s) of $\mathbf{x}$. This can destabilize population codes by causing many cells to track similar features, but can be stabilized with response normalization, which we will illustrate later in a nonlinear network.

Recurrence in a linear model Hebbian homeostasis improves stability, but does not stabilize the population code in the long-term, since the tuning of each neuron can diffuse slowly. Recurrent dynamics address this by deleting changes in $\mathbf{y}(\theta)$ that are not consistent with the learned structure of θ. We define recurrent weights $\mathbf{R}$ that transform the feed-forward activations $\mathbf{y}_f = \mathbf{W}^\top \mathbf{x}$ into an error-corrected training signal $\mathbf{y}_r = \mathbf{R}^\top \mathbf{y}_f$. This gives a new Hebbian learning term that cancels the difference between feed-forward and recurrent activity:

$$\Delta \mathbf{W} \propto \gamma [\langle \mathbf{x} \mathbf{y}_r^\top \rangle - \mathbf{w}_t] + \rho \langle \mathbf{x} (\mathbf{y}_r - \mathbf{y}_f)^\top \rangle, \tag{13}$$

where ρ sets the influence of recurrent dynamics on the decoding weights. The error signal $\mathbf{y}_r - \mathbf{y}_f$ can be computed using recurrent negative feedback in a predictive coding framework (Methods: *Linear network with recurrence*). The benefits of recurrence in a linear network are limited (Fig. 3d). The benefits of recurrence in nonlinear networks are more substantial, as we shall see later (Fig. 4).

If drift is random, there is a small chance it will disrupt the low-dimensional manifold that encodes θ. Downstream neurons cannot detect this, and these small errors accumulate. Noise in activity and learning rules would cause a similar effect. These errors scale inversely with the ratio of the population size to the dimensionality of θ. Accordingly, code stability grows linearly with network redundancy (Fig. 3e).

Overall, the linear model provides important intuition: Hebbian homeostasis is an inevitable consequence of the interaction between Hebbian learning and homeostatic processes in single cells. This stabilizes neural function in the presence of drift; Recurrent dynamics can provide further stability (Fig. 3cd). As we discuss next, other constraints, such as nonlinear recurrent dynamics and response normalization, can also confer marked stability.

2.5 Nonlinearity and response normalization

Much of the intuition from the linear network extends to the nonlinear case. We assume that neuronal responses are (approximately) locally linear, so the same Hebbian learning rules apply. However, a nonlinear network has key advantages: It is better at correcting errors, and it lets us examine the effect of response normalization on readout stability.

Response normalization controls the average firing rate in a local population of neurons. It is supported experimentally, and implicated in diverse sensory computations (for review, see [92]). Importantly, response normalization causes neurons in a local population to compete to remain active. Competition can encourage neurons to acquire diverse tunings, forming a

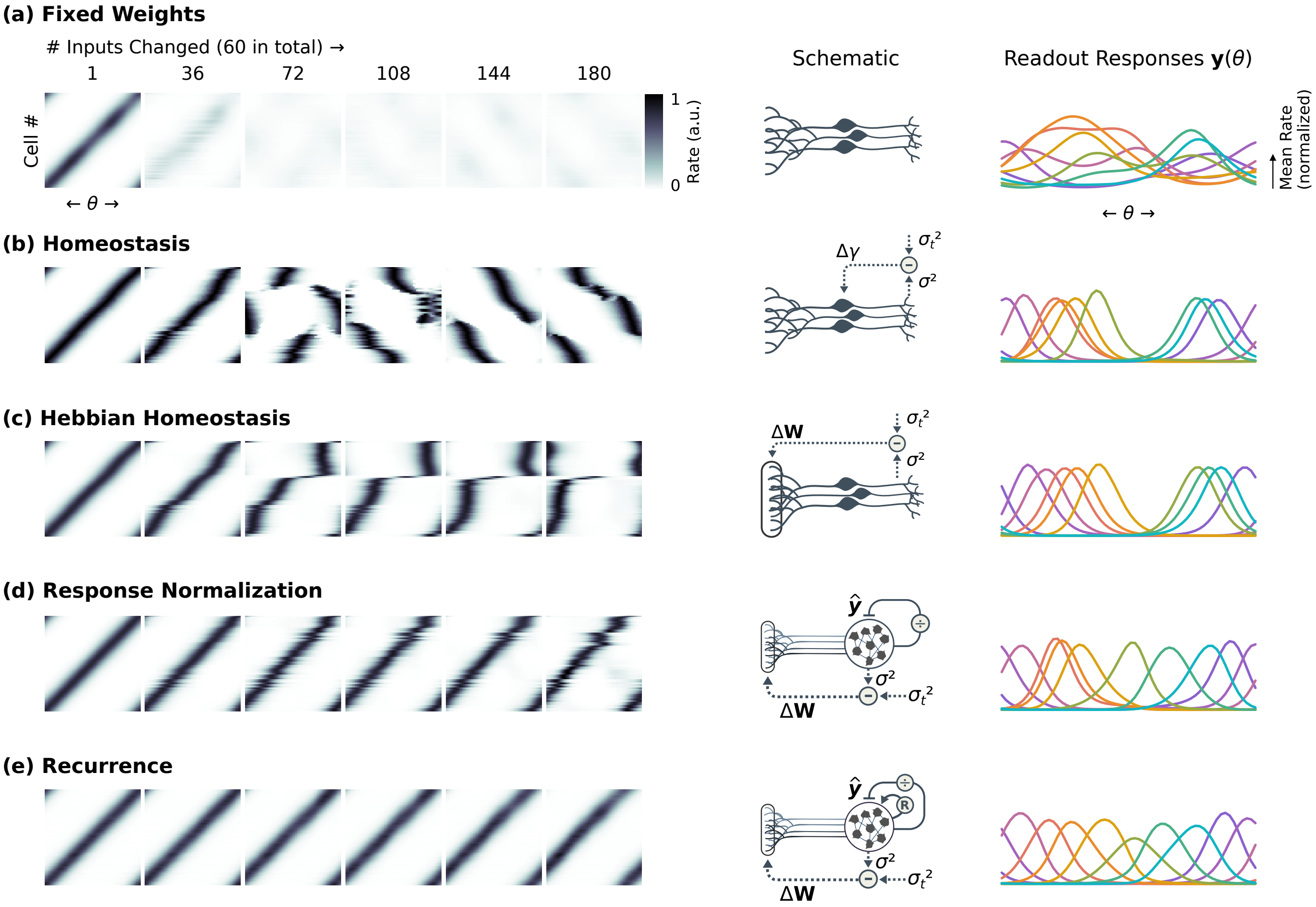

Figure 4: ***Self-healing readout in a nonlinear rate network.*** Each plot shows (left) the stability of a population readout $\mathbf{y}(\theta)$ from a drifting code $\mathbf{x}(\theta)$ over time, (middle) a schematic of the readout dynamics, and (right) a plot of select readout unit's tuning to θ if 55 out of 60 (92%) of the encoding cells were to abruptly switch to a new, random tuning. **(a)** For fixed readouts, representational drift in the encoding population gradually destroys the feature conjunctions used to define selective activity in the readout. **(b)** Homeostatic processes could stabilize the mean firing rate and variability in readout cells. For small amounts of drift, homeostasis can compensate for loss of drive. However, drift eventually disrupts the readout's tuning curve. **(c)** Hebbian homeostasis can preserve the statistics of tuning curves in single cells, by using a neuron's own output as a training signal to update decoding weights. However, this process is not lossless, and the population code in the readout degrades over time. **(d)** Response normalization controls the population firing rate, causing neurons to compete for activation. This stabilizes the statistics of the population code, but readout neurons can still swap preferred tunings, degrading the readout. **(e)** Recurrent activity, in which the network predicts its own activity, can enforce population correlations. This limits the structure of the readout to the ring-like encoding in which it was first trained. Here, the only drift that is permitted is along the symmetry of the circular state θ.

population of localized receptive fields that tile the encoded latent variable space [105–107].

Indeed, without some additional mechanisms to retain diverse population tuning curves, the Hebbian homeostasis rules derived so far will eventually cause the population readout to collapse toward the subspace with the most variability. Recurrent connections and response normalization can prevent this. For a thorough exploration of what this might look like in linear and rectified linear models, see Sengupta et al. [105], Pehlevan et al. [108, 109, 110], Lipshutz et al. [111]. Briefly, the idea is this: neurons learn to explain the latent causes of their inputs. In linear models, this amounts to finding the principle components or principle subspace [110, 111]. However, neurons must coordinate to ensure that they don't all focus on the same direction of maximum variability. Response normalization, and other forms of competition, ensure full coverage of the encoding space. In models with a rectifying nonlinearity, this leads to the bump-like tuning curves that tile the encoding space [105, 112].

Nonlinear, recurrent networks require specific architectural details to avoid unstable dynamics. To avoid this complexity, we model recurrent dynamics and response normalization as discrete transformations. For response normalization, we divide the rates by the average firing rate across the population $\langle \mathbf{y}_f(\theta) \rangle$:

$$\mathbf{y}_d(\theta) = \mathbf{y}_f(\theta) / \langle \mathbf{y}_f(\theta) \rangle \cdot \mu_p, \qquad (14)$$

where μ_p is the target average firing rate across the population. For recurrent connections, we train the readout to predict its own activity using fixed set of recurrent weights $\mathbf{R}$:

$$\mathbf{y}_r(\theta) = \phi[\mathbf{R}^\top \mathbf{y}_d(\theta)] \qquad (15)$$

This signal $\mathbf{y}_r(\theta)$ can be used as a training signal to continuously update the forward encoding weights, as in (5).

Figure 4 summarizes the influence of response normalization and recurrent dynamics in a nonlinear readout. As in the linear case, fixed weights are unstable. Classical homeostasis provides only short term stability. Hebbian homeostasis stabilizes tuning curve statistics, but does not prevent eventual collapse of the population code (Fig. 4a-c).

Perhaps surprisingly, response normalization alone improves stability substantially (Fig. 4d). This is because it creates repulsive force between their preferred tunings under the influence of Hebbian plasticity. For the one-dimensional manifold θ explored here, this repulsion constrains the possible rearrangements. Drift must be large enough to cause two readout neurons to exchange their preferred tunings, to induce any long-term changes. Note that tuning curves would be much less constrained in higher dimensional spaces, and we should expect the stabilizing effect of crowding to diminish for higher-dimensional θ.

With recurrent dynamics, the nonlinear readout is exceptionally stable (Fig. 4d). The recurrent weights strongly constrain the correlated activity patterns in $\mathbf{y}(\theta)$, and suppressing any activity that does not match the ring structure learned initially. Drift can only occur along directions of symmetry in the underlying encoded space θ. In this case, θ is a circular variable, so drift can rotate the readout, but no other changes are permitted. This illustrates that internal predictive models can strongly constrain network activity, and that these constraints allow populations of neurons to tolerate complete reconfiguration of the inputs that drive them.

3 Discussion

In this work, we outlined homeostatic principles that could allow stable and plastic representations to coexist in the brain. We argue that self-healing codes should have of three components: (I) Neurons codes should be robust to drift; (II) Neurons should use their own output as a training signal to update their decoding weights, and (III) Stable neural codes should homeostatically preserve internal models, which are reflected in stable population statistics.

Here, we considered two populations, one stable and one unstable. This could reflect communication between stable and plastic components of the brain, or the interaction between stable and plastic neurons within the same neural population. This is consistent with experiments that find consolidated stable representations [113], and with the view that neural populations contain a mixture of stable and unstable cells [114, 115].

However, there is no requirement that a neuron that is stable at present must remain so. Over time, neurons could enter or leave this stable core. As long as some stable neurons remain, long-term representations could persist. This implies a general principle that supports reallocation of the function of single neurons, while preserving internal models. It also raises the question of whether a stable population is even necessary: could functional stability be achieved by several plastic populations tracking each-other? This points to a potentially powerful generalization of homeostatic principles, which could explain the long-term robustness of distributed neural representations.

Here, we considered how networks might stabilize a pre-existing trained structure. How are these stable representations learned? Once learned, can they be updated? A crucial assumption in our work is that neurons generate their own internal training signals. For single cells, this amounts to error correcting across the pool of its own synaptic inputs. For networks, this corresponds to prediction errors coming from recurrent or top-down dynamics. These error signals are precisely the same ones that would be used for learning from external error feedback. During learning, recurrent and top-down prediction errors propagate high-level reinforcement signals back to local neural populations [47]. These prediction errors are carried by the same mechanisms that we use here to achieve homeostasis. Hebbian homeostasis, then, can be viewed as a natural consequence of predictive learning mechanisms in the absence of external error feedback.

The brain supports both consolidated and volatile representations, respectively associated with memory and learning [5, 114, 116–122]. Artificial neural networks have so far failed to imitate this, and suffer from catastrophic forgetting wherein new learning erases previously learned representation [123]. Many strategies have been proposed to mitigate this (e.g. [124–127]). Broadly, all of these methods segregate stable and unstable representations into distinct subspaces of the possible synaptic weight changes. These learning rules therefore amount to preventing disruptive drift in the first place.

The strategies we explore here are fundamentally different. We do not restrict changes in weights or activity: the encoding population is free to reconfigure arbitrarily. However, any change in a neural code leads to an equal and opposite change in how that code is interpreted—The brain must publish new translations of its changing internal language. This constraint preserves the functional relationships between neural populations.

The approach developed here shares some similarities with approaches to attenuate forgetting using replay during sleep, or the equivalent in artificial networks (e.g. [128–131]). The internal models must be occasionally re-activated through either rehearsal or replay, in order to detect and correct inconsistencies caused by drift. If this process occurs too infrequently, drift becomes large, and the error correction will fail.

Here, we focused on homeostatic maintenance of function despite drifting population codes. It is worth exploring whether a similar process can explain how the brain preserves learned representations despite neuronal death. In developmental pruning, the brain removes synapses and neurons without loss of function [132–134]. Existing models of pruning require ongoing learning to prevent loss of learned representations [135–137]. Homeostatic preservation of predictive models may allow the brain to benefit from large networks during learning [138–140], and optimize these networks without extensive re-training. The processes we examine here may also be similar to those that allow transfer of learned motor skills despite gradual change in the readout of a brain-machine interface [141–143].

To integrate stable and plastic representations, changes anywhere in the brain must be accompanied by compensatory changes throughout the brain. The learning rules we explored here placed a particular emphasis on Hebbian homeostasis, and the role of predictive coding in generating robust representations. In the long term, these processes could support widespread reallocation or reconsolidation of neuronal function. Further exploration of these principles may clarify how the brain can be simultaneously plastic and stable, and provide clues to how to build artificial networks that share these properties.

4 Methods

4.1 Data and analysis

Data shown in Figure 1b,c were taken from Driscoll et al. [6], and are available online at at Dryad [144]. Examples of tuning curve drift were taken from mouse four, which tracked a subpopulation of cells for over a month. Normalized dF/F calcium transients were band-pass filtered between 0.3 and 3 Hz, and individual trial runs through the T maze were extracted. Calcium fluorescence traces from select cells were aligned based on task pseudotime (0: start, 1: reward). The activity of each cell was z-scored within each trial to yield a normalized log-fluorescence signal. On each day, normalized log-fluorescence was averaged over all trials and then exponentiated to generate the average tuning curves shown in Figure 1b. for Figure 1c, a sub-population of forty cells was selected at random, and sorted based on their peak firing location on the first day. For further details, see [6, 19].

4.2 Simulated drift

We sample a random walk on encoding weights $\mathbf{U}$ as an Ornstein Uhlenbeck (OU) process with unit steady-state variance and time constant τ, measured in days. Given τ, and the constraint that the steady-state variance of an OU process is $\frac{1}{2}\tau\sigma^2 = 1$, we set the noise variance to $\sigma^2 = 2/\tau$. In discrete time this is sampled with $\alpha = \sigma^2\Delta t$:

$$u_{ij}^{t+1} = u_{ij}^{t}\sqrt{1-\alpha} + \sqrt{\alpha}\xi, \qquad \xi\sim\mathcal{N}(0,1) \tag{16}$$

This yields an embedding of θ in the activity of the N-dimensional encoding population $\mathcal{X}$ that changes gradually and randomly over time. The structure of θ encoded in $\mathbf{s}(\theta)$ is inherited by $\mathbf{a}(\theta) = \mathbf{U}^\top\mathbf{s}(\theta)$.

This model preserves the amount of population variability in $\mathbf{a}(\theta)$ driven by θ, in expectation:

$$\left\langle \|\nabla_\theta \mathbf{a}(\theta,t)\|^2\right\rangle = N\cdot\mathrm{tr}[\nabla_\theta\Sigma(\theta,\theta')\nabla_{\theta'}^\top] = N\cdot\|\nabla_\theta\mathbf{s}(\theta)^\top\|^2 \tag{17}$$

In the special case of an exponential nonlinearity ϕ= exp, the trace of Fisher information of $\mathbf{x}(\theta)$ is proportional to the average variation in $\mathbf{a}(\theta)$ driven by θ:

$$\mathrm{tr}[\mathcal{I}(\theta)] \propto \left\langle\|\nabla_\theta \ln[\mathbf{x}(\theta,t)]\|^2\right\rangle = \left\langle\|\nabla_\theta\mathbf{a}(\theta,t)\|^2\right\rangle \tag{18}$$

Formally, the Fisher information is infinite when the noise in $\mathbf{x}$ is zero, but (18) can be viewed as the zero-variance limit of homogeneous and i.i.d. Gaussian noise with suitable normalization.

In expectation then, this random walk in the encoding weight space preserves the overall population code statistics: It preserves the geometry of θ in the correlations of $\mathbf{a}(\theta)$, and the average amount of information about θ encoded in the population activations.

4.3 Weight filtering in a linear model

We consider a linear version of our encoding-decoding model (Eqns. 2-6), whose weights and activity change across days ("d")

$$\begin{aligned}\mathbf{x}_d(\theta) &= \mathbf{U}_d^\top\mathbf{s}(\theta)\\ \mathbf{y}_d(\theta) &= \mathbf{W}_d^\top\mathbf{x}_d(\theta)\end{aligned} \tag{19}$$

Drift can be viewed as a slow-timescale component of noise, and a readout that is robust to noise can also tolerate some amount of drift. Denote the drift in the code between days as $\Delta\mathbf{x}(\theta)$, and assume that it can be modeled as Gaussian:

$$\Delta\mathbf{x}(\theta) \sim \mathcal{N}(0,\Sigma_\Delta) \tag{20}$$

This Gaussian model captures diffusive drift like the OU process (16) introduced earlier. For training signals $(\mathbf{x}_0, \mathbf{y}_0^*)$, the least-squares optimal weights for day d+1 trained on activity on day d is given by regularized linear regression:

$$\mathbf{W}_{d+1} = [\Sigma_d + \Sigma_\Delta]^{-1}\Sigma_{0,\mathbf{y}_0^*} \tag{21}$$

where Σ_d is the covariance of $\mathbf{x}_d(\theta)$, and $\Sigma_{0,\mathbf{y}_0^*}$ is the cross covariance between the encoding population activity and the target readout tuning curves $\mathbf{y}_0^*$.

We needn't estimate these regularized weights from scratch. If we have already weights $\mathbf{W}_d$ trained on day d, then we can prepare regularized weights for the subsequent day $\mathbf{W}_{d+1}$ by updating these existing weights. This also realigns the decoding weights with the correlation structure of the current encoding, Σ_d=$\langle \mathbf{x}_d \mathbf{x}_d{}^\top \rangle$:

$$\mathbf{W}_{d+1} = [\Sigma_d + \Sigma_\Delta]^{-1}\Sigma_d \mathbf{W}_d. \tag{22}$$

(c.f. Eq. 9, Results.) This is equivalent to using the activity on the current day, $\mathbf{x}_{d+1}$, to predict the corresponding activity on the previous day $\mathbf{x}_d$:

$$\hat{\mathbf{x}}_d = \Sigma_d[\Sigma_d + \Sigma_\Delta]^{-1}\mathbf{x}_{d+1} \tag{23}$$

Applying (23) iteratively yields an estimate of the original code $\hat{\mathbf{x}}_0$, thereby translating the current representation $\mathbf{x}_d$ back in time to when the readout was first learned:

$$\hat{\mathbf{y}}(\theta) = \mathbf{W}_0^\top \left\{\textstyle\prod_{d' \in 0..d-1} \Sigma_{d'}[\Sigma_{d'} + \Sigma_\Delta]^{-1}\right\} \mathbf{x}_d(\theta). \tag{24}$$

Since the readout activity is driven by these decoding weights, $\mathbf{y}_d = \mathbf{W}_d^\top \mathbf{x}_d$, this recursive filtering can be interpreted by the network re-training itself using its own output:

$$\begin{aligned} \mathbf{y}^* &= \mathbf{W}_d^\top \mathbf{x}_d \\ \mathbf{W}_{d+1} &= [\Sigma_d + \Sigma_\Delta]^{-1}\Sigma_{d,\mathbf{y}^*} \\ \Sigma_{d,\mathbf{y}^*} &= \langle \mathbf{x}_d \mathbf{y}^{*\top} \rangle = \langle \mathbf{x}_d \mathbf{x}_d{}^\top \mathbf{W}_d \rangle = \Sigma_d \mathbf{W}_d \\ \Rightarrow \mathbf{W}_{d+1} &= [\Sigma_d + \Sigma_\Delta]^{-1}\Sigma_d \mathbf{W}_d \qquad \text{[c.f. Eqs. 9,22]} \end{aligned} \tag{25}$$

To summarize, tracking an unstable code involves filtering the current code-words $\mathbf{x}_d$ to recover the original code $\mathbf{x}_0$ against which the readout was first trained. In a linear, Gaussian model, this can be computed by iteratively re-training the decoding weights using the network's own output.

The linear Bayesian model in Equations (19)-(25) incorporate the assumption that the encoding $\mathbf{x}$ changes, but not that $\Pr(\theta)$ and the primary inputs $\mathbf{s}(\theta)$ are fixed. How might neurons incorporate this? The readout population cannot access $\mathbf{s}(\theta)$, but it could measure its own statistics:

$$\Pr(\mathbf{y}) = \textstyle\int \mathbf{y}(\theta)\Pr(\theta)\, d\theta. \tag{26}$$

For example, in the linear model (Section 2.4), $\mathbf{y}(\theta)$ is a zero-mean Gaussian variable, so $\Pr(\mathbf{y})$ is encoded entirely in the covariance $\Sigma_\mathbf{y}$:

$$\Sigma_\mathbf{y} = \langle \mathbf{y}\mathbf{y}^\top \rangle = \textstyle\int \mathbf{y}(\theta)\mathbf{y}(\theta)^\top \Pr(\theta) d\theta \tag{27}$$

Since $\Sigma_\mathbf{y}$ is inherited from $\Pr(\mathbf{s}(\theta))$, stable readouts must exhibit stable $\Sigma_\mathbf{y}$. The converse is not true, but is a useful constraint that can improve stability. This covariance is readily accessible: its diagonal is simply the firing rate variability of single neurons, and its off-diagonal terms can be encoded in recurrent connections that constrain population activity.

4.4 Synaptic learning rules

The homeostatic learning rule (10) is simple, but unrealistic: it requires tracking the covariance of the encoding population, and solving a linear system by matrix inversion. Neither of these are things that single neurons could do. However, these operations are equivalent to linear regression, which can be computed in an online manner using stochastic gradient descent.

Least Mean Squares (LMS; [145, 146]) is an online stochastic gradient descent algorithm that solves the linear regression problem $\mathbf{y}$=$\mathbf{W}^\top\mathbf{x}$, converging (with noise) to the solution $\mathbf{W}$=$\Sigma_\mathbf{x}{}^{-1}\Sigma_{\mathbf{xy}}$, by minimizing the following objective via stochastic gradient descent:

$$\mathbf{w} = \underset{\mathbf{w}}{\operatorname{argmin}} \tfrac{1}{2} \left\langle \|\mathbf{w}^\top\mathbf{x} - \mathbf{y}\|^2 \right\rangle \tag{28}$$

Given a single observation $(\mathbf{x}_t, \mathbf{y}_t)$ at time t, LMS computes the following online weight update:

$$\begin{aligned} \Delta\mathbf{w}_t &\propto -\nabla_{\mathbf{w}_t} \tfrac{1}{2} \left\langle \|\mathbf{w}_t^\top\mathbf{x} - \mathbf{y}\|^2 \right\rangle \\ &= \Sigma_{\mathbf{x},\mathbf{y}} - \Sigma_\mathbf{x}\mathbf{w}_t \\ &\approx \mathbf{x}_t\mathbf{y}_t{}^\top - \mathbf{x}_t\mathbf{x}_t{}^\top\mathbf{w}_t \\ &= \mathbf{x}_t(\mathbf{y}_t - \mathbf{w}_t^\top\mathbf{x}_t)^\top. \end{aligned} \tag{29}$$

Recall the formula for the filtering weight update, with homeostatic gain re-scaling of g=$\sigma_0/\sigma_{y\cdot}$:

$$\mathbf{w}_{d+1} = g \cdot [\Sigma_d + \Sigma_\Delta]^{-1}\Sigma_d \mathbf{w}_d \tag{30}$$

This is a batched update, which uses activity on a given day to update the weights for the following day. It minimizes the following objective:

$$\begin{aligned} \mathbf{y} &= g \cdot \mathbf{w}^\top\mathbf{x} \\ \mathbf{w} &= \underset{\mathbf{w}}{\operatorname{argmin}}\ \tfrac{1}{2} \left\{ \left\langle \|\mathbf{w}^\top\mathbf{x} - \mathbf{y}\|^2 \right\rangle + \mathbf{w}^\top\Sigma_\Delta\mathbf{w} \right\} \\ &= \underset{\mathbf{w}}{\operatorname{argmin}}\ \tfrac{1}{2}\, \mathbf{w}^\top \left\{ (1-g^2)\Sigma_d + \Sigma_\Delta \right\} \mathbf{w} \end{aligned} \tag{31}$$

In the online model, we treat drift as occurring gradually and continuously, over small intervals Δt. The incremental drift is therefore $\Delta t \cdot \Sigma_\Delta$, and the homeostatic gain adjustments are small, $g^2 \approx 1 + \gamma\Delta t$. The weight update (30) for a self-healing code is also a linear least-squares problem. In analogy to LMS, an online stochastic gradient solution for the self-healing weight update rule is:

$$\begin{aligned} \tfrac{\Delta\mathbf{w}_t}{\Delta t} &\propto -\tfrac{1}{\Delta t} \left[(1-g^2)\Sigma_d + \Delta t \cdot \Sigma_\Delta \right] \mathbf{w}_t \\ &\approx [\gamma\, \mathbf{x}_t\mathbf{x}_t^\top - \Sigma_\Delta]\mathbf{w}_t \\ &= \gamma\, \mathbf{x}_t\mathbf{y}_t^\top - \Sigma_\Delta\mathbf{w}_t \qquad \text{[c.f. Eq. (11)]} \end{aligned} \tag{32}$$

This reduces to the Hebbian homeostatic weight update, (7), with γ=ε_σ providing negative feedback to stabilize the neuron's firing-rate variability. (32) also contains an extra term, $-\Sigma_\Delta\mathbf{w}_t$, which acts as regularizing weight decay. The drift Σ_Δ could be estimated in several ways. It might simply be initialized heuristically as a constant weight decay $\Sigma_\Delta \propto I$. It is also possible to use changes in neuronal variability as a proxy for drift.

4.5 Estimating the rate of drift

Empirically, we observe that the population statistics for the tuning curves $\mathbf{x}(\theta)$ are stable despite drift [6]. The tuning curve of each encoding cell $x(\theta)$ can be viewed as a vector in this space of possible tuning curves. For large populations, the total amount of task-related variability is approximately conserved. This implies that drift is, on average, mostly rotational. If rotational drift rotates our code by amount ϕ away from the subspace spanned by our current decoding weights, it will lead to a loss of drive to the readout neurons, which is approximately $\cos^2(\phi)$.

The homeostatic gain adjustment acts based on the observed loss of drive. Assuming our target variance is one, σ_t^2=1, a variability decreases of $\cos^2(\phi)$ requires a gain adjustment of g=$1/\cos(\phi)$. For small amounts of drift, a first-order Taylor expansion yields $\gamma \approx \phi^2$. The current value γ is therefore also an estimate of the drift rate, and one may write:

$$\begin{aligned}\hat{\Sigma}_\Delta &\approx \gamma I \\ \tfrac{\Delta \mathbf{w}_t}{\Delta t} &\approx \gamma \cdot [\mathbf{x}_t \mathbf{x}_t^\top - I]\mathbf{w}_t\end{aligned} \tag{33}$$

Another way to arrive at (33) is to assume that drift (and therefore any compensatory weight changes) should be tangent to the current decoding weight vector (Figure 3a). This has an intuitive interpretation: if we assume that the encoding of θ is stable over time *at the population level*, then we know that there is always *some* linear combination of decoding weights that can read out a target tuning curve $y(\theta)$ from $\mathbf{x}(\theta)$. That is, the overall statistics of the weight vector should also be stable. Drift causes these decoding weights to point in a slightly different direction. Tracking drift therefore amounts to rotating the weight vector to point in this new direction. Large reconfigurations of the encoding space can therefore be tracked if drift is gradual (Figure 3b).

To ensure that gain homeostasis can converge in the absence of drift, one might use a faster learning rate η_γ>1 for gain adjustment, which amounts to:

$$\tfrac{\Delta \mathbf{w}_t}{\Delta t} \approx \gamma \cdot [\eta_\gamma \mathbf{x}_t \mathbf{x}_t^\top - I]\mathbf{w}_t \tag{34}$$

4.6 Linear network with recurrence

So far, we have explored self-healing codes in the case of a single neuron, which uses a measurement of its own variability to detect and correct for drift. One way to extend this to populations is to assume that the activity in the readout, $\mathcal{Y}$, is constrained by local recurrent connections. This recurrent activity provides additional error correction [65]. In this scenario, the decoding weights and recurrent connections incorporate the prior knowledge that Σ_y should remain stable over time.

A simple version of this mechanism might use feed-forward activity $\mathbf{y}_f$=$\mathbf{W}^\top\mathbf{x}$ to generate regularized predictions $\mathbf{y}_r$. This regularized estimate might be computed via local, recurrent weights $\mathbf{R}$ that encode a fixed prior model of Σ_y:

$$\begin{aligned}\mathbf{y}_f &= \mathbf{W}^\top\mathbf{x} \\ \mathbf{y}_r &= \mathbf{R}^\top\mathbf{y}_f \\ \mathbf{R} &= [\Sigma_\mathrm{y} + \kappa I]^{-1}\Sigma_\mathrm{y},\end{aligned} \tag{35}$$

where κ sets the strength of the regularization in the recurrent dynamics.

This pools information across the readout population by linearly predicting the readout's activity from itself, with regularization strength α. This can also be viewed as Gaussian process (GP) smoothing, where Σ_y encodes the GP prior kernel using the "true" tuning curves $\mathbf{y}(\theta)$ to support the function space. (35) can be computed as a steady-state solution of a recurrent network that computes a prediction error $\mathbf{W}^\top\mathbf{x} - \mathbf{y}$ using inhibitory feedback:

$$\tau\dot{\mathbf{y}} = -y + \tau\Sigma_\mathrm{y}[\mathbf{W}^\top\mathbf{x} - \mathrm{y}], \tag{36}$$

where $\tau = 1/\kappa$. If $\mathbf{x}$ varies slowly relative to the time constant τ, and if (36) converges, then it converges to (35), and tracks $\mathbf{y}_r(t)$. We stop short of specifying a specific biological realization of (36), but this feedback-based solution is consistent with the prevailing theory that the brain learns and computes using prediction errors [96, 99, 147].

Recurrent feedback yields a new error signal, $\mathbf{y}_r$−$\mathbf{y}_f$ that detects when the decoded activity strays outside of the low-dimensional subspace of the initial code, $\mathbf{y}_0(\theta)$. This error can be added to the weight update (34) to yield a combined update that reflects two constraints: Hebbian homeostasis, and local recurrent dynamics (Results, Eq. 13).

In this form, it becomes clear that the recurrent dynamics in $\mathbf{y}(\theta)$ truly are predictive dynamics. A Hebbian rule which tracks drift is, essentially, minimizing the errors in the online predictions that $\mathbf{y}$ makes about the activity $\mathbf{x}$. In this paper, we consider only the case where $\mathbf{x}$ changes so slowly that this prediction should be the identity map. However, in a scenario where $\mathbf{x}$ has nontrivial temporal dynamics, such recurrent computations and learning inherently learn an asymmetric model that captures how θ evolves in time.

4.7 Linear simulations

We simulated a self-healing linear network encoding a circular latent variable $\theta \in [0, 2\pi)$, discretized into L=60 spatial bins. We sampled K=200 randomly-drifting spatial features $\mathbf{x}(\theta)$ from a Gaussian process on θ, with an exponentiated quadratic (i.e. radial basis; Gaussian) covariance kernel with a spatial standard deviation of σ_l=9 bins, scaled so that the standard deviation of each feature was s=0.15. These features underwent Ornstein Uhlenbeck drift over time, with a time-constant of τ=50 days (Eq. 16). M=50 readout units $\mathbf{y}(\theta)$ were initialized with bump-like tuning curves, modeled as Gaussians with σ_y=9 bins, evenly distributed over a range of preferred tunings θ_0. These readouts were given a homeostatic target variance of σ_t=1.

We simulated 500 days of drift—ten times of the correlation time for the drifting encoding features. This allowed multiple complete reconfigurations of the encoding population. We simulated Hebbian homeostasis using a continuous-time learning rule (Eq. 34) applied for 500 time-steps on each day, with a learning rate of $1{\times}10^{-5}$ per time-step. These updates were batched, so that rather than sampling individual stimuli and using $\mathbf{x}\mathbf{x}^\top\mathbf{W}$ to calculate updates in stochastic gradient descent, we directly apply the expectation $\Sigma_\mathbf{x}\mathbf{W}$.

We evaluated three scenarios: fixed weights, Hebbian homeostasis, and Hebbian homeostasis with recurrent prediction errors (Results, section 2.4, and Figure 3). We modeled recurrence as an additional linear map $\mathbf{y}_r$=$\mathbf{R}^\top\mathbf{y}_f$ as in Equation (35), and the resulting $\mathbf{y}_r$ was used as a training signal in a batched least-mean-squares continuous-time gradient update (Eq. 29). To summarize the relative performance of these three scenarios (Fig. 3d), we sampled 20 random realizations of the aforementioned simulations.

The ability of the linear model to error-correct is limited by the amount of drift that projects onto the low-dimensional subspace in $\mathbf{x}(\theta)$ that encodes θ. While the total amount of drift increases for larger populations, averaging predicts that the disruptive effect of drift (in terms of squared error) should scale inversely with population size. To verify this, we simulated a range of models with different degrees of redundancy. We simplified the input features $\mathbf{s}(\theta)$ to reflect a K-dimensional Gaussian variable θ, encoded in an $N > K$ population. The readout $\mathbf{y}(\theta)$ was initialized to recover θ via linear regression. As above, we simulated 500 days of random drift as an O.U. process on the encoding weights, using Hebbian homeostasis (without recurrence). For each network realization, we sampled ten instances of the initial features and network, and then five independent realizations of random drift for each instance.

4.8 Nonlinear simulations

For the nonlinear readout, we simulated a circular variable $\theta{\in}[0, 2\pi)$ divided into L=60 discrete bins. We sampled K=60 features $\mathbf{s}(\theta)$ from a Gaussian process on θ, with zero mean and an exponentiated quadratic covariance kernel with standard deviation σ_l=15. We allowed individual encoding units $x_n(\theta)$ to change abruptly, rather than undergo a continuous random walk. We did this by re-sampling features one-at-a-time, and running Hebbian homeostasis each time 8% of the encoding features changed. This approach emphasized that the nonlinear readout can track drift through multiple complete reconfigurations of the encoding population. Encoding features were normalized to range from 0 to 1, then passed through a nonlinearity $\mathbf{x}(\theta){=}\exp[z(\theta){-}\tfrac{1}{2}]$ to simulate sparse, non-negative network inputs.

We initialized N=60 linear-nonlinear readout neurons ($\mathbf{y}(\theta)$; Eq. 2) with Gaussian tuning curves $\mathbf{y}_0(\theta)$ with standard deviation σ_y=5 bins, and with preferred tunings θ_0 evenly distributed on $[0, 2\pi)$. Readout weights $\mathbf{W}$ were trained via gradient descent to minimize a loss similar to a log-linear Poisson model. This incorporates the constraint that tunings can be matched less precisely at higher firing rates:

$$\mathbf{W} = \underset{\mathbf{W}}{\operatorname{argmin}} \left\langle \exp[\mathbf{W}^\top \mathbf{x}] - \mathbf{y}_0 \circ \mathbf{W}^\top \mathbf{x} \right\rangle + \kappa \|\mathbf{W}\|^2, \tag{37}$$

where $\circ$ denotes element-wise multiplication, the expectation $\langle\cdot\rangle$ is taken over θ and the readout population, and the regularization strength is κ=10^{-2}.

The homeostatic set-points for the mean and the variance of the firing rate, μ_t and σ_t^2, were taken from the statistics of these initial tuning curves. We implemented Hebbian homeostasis by defining slow variables γ and β, which track the deviations of the neuron's firing rate statistics from its homeostatic set points. Weights were trained to restore these set-points via a continuous-time Hebbian learning rule (Eq. 7). 50 iterations of this learning rule were applied each time 8% (5 out of 60) of the encoding population had reconfigured.

For nonlinear neurons, homeostasis of the mean-rate and variability interact. Controlling the variability can change the overall excitability of the neuron, and can lead to instability. To address this, we used different learning rates η_β=0.9 and η_σ=0.1 for the mean-rate and variability, respectively. The more rapid control of mean-rate avoids instability related to variability homeostasis.

To simulate response normalization, we divided the response $\mathbf{y}(\theta)$ by the average population rate, scaled to preserve the population rates seen in the initial network configuration, as in Equation (14). To model recurrent dynamics, we trained another set of fixed recurrent weights $\mathbf{R}$ as in Equation (15), with a gradient descent objective similar to the one used to initialize the decoding weights (Eq. 37).

$$\mathbf{R} = \underset{\mathbf{R}}{\operatorname{argmin}} \left\langle \exp[\mathbf{R}^\top \mathbf{y}_0] - \mathbf{y}_0 \circ \mathbf{R}^\top \mathbf{y}_0 \right\rangle, +\kappa_r \|\mathbf{R}\|^2 \tag{38}$$

with regularization strength of κ_r=10^{-4}.

These recurrent predictions were applied to generate a revised prediction $\mathbf{y}_r(\theta)$ after applying response normalization. For both response normalization and the recurrent model, "error-corrected" estimates $\hat{\mathbf{y}}$=$\mathbf{y}_d$ or $\hat{\mathbf{y}}$=$\mathbf{y}_r$ were used to retrain the decoding weights via Hebbian learning, with regularizing weight decay rate of ρ_d=$\tfrac{1}{3}{\times}10^{-3}$:

$$\begin{aligned} \mathbf{y}_f &= \exp(\mathbf{U}^\top \mathbf{x}) \\ \Delta\mathbf{U} &= \eta \left\langle \mathbf{x}[\hat{\mathbf{y}} - \mathbf{y}_f]^\top \right\rangle - \rho_d \mathbf{U}, \end{aligned} \tag{39}$$

with a learning rate of η=0.5. Note that the above (39) corresponds to online gradient descent of an objective similar to those used to train the initial forward and recurrent weights (Eqs. 37, 38).

5 Acknowledgments

This work was supported by ERC grant StG 716643 FLEXNEURO and HFSP grant RGY0069.

References

[1] Simon Rumpel and Jochen Triesch. The dynamic connectome. *e-Neuroforum*, 22(3):48–53, 2016.

[2] Anna R Chambers and Simon Rumpel. A stable brain from unstable components: emerging concepts and implications for neural computation. *Neuroscience*, 357: 172–184, 2017.

[3] Gianluigi Mongillo, Simon Rumpel, and Yonatan Loewenstein. Intrinsic volatility of synaptic connections—a challenge to the synaptic trace theory of memory. *Current opinion in neurobiology*, 46:7–13, 2017.

[4] Yaniv Ziv, Laurie D Burns, Eric D Cocker, Elizabeth O Hamel, Kunal K Ghosh, Lacey J Kitch, Abbas El Gamal, and Mark J Schnitzer. Long-term dynamics of ca1 hippocampal place codes. *Nature neuroscience*, 16(3):264, 2013.

[5] Samuel Jordan Levy, Nathaniel Reid Kinsky, William Mau, David W Sullivan, and Michael E Hasselmo. Hippocampal spatial memory representations in mice are heterogeneously stable. *bioRxiv*, page 843037, 2019.

[6] Laura N Driscoll, Noah L Pettit, Matthias Minderer, Selmaan N Chettih, and Christopher D Harvey. Dynamic reorganization of neuronal activity patterns in parietal cortex. *Cell*, 170(5):986–999, 2017.

[7] Abhinav Singh, Adrien Peyrache, and Mark D Humphries. Medial prefrontal cortex population activity is plastic irrespective of learning. *Journal of Neuroscience*, 39(18):3470–3483, 2019.

[8] Benjamin R Cowley, Adam C Snyder, Katerina Acar, Ryan C Williamson, M Yu Byron, and Matthew A Smith. Slow drift of neural activity as a signature of impulsivity in macaque visual and prefrontal cortex. *Neuron*, 108 (3):551–567, 2020.

[9] Daniel Deitch, Alon Rubin, and Yaniv Ziv. Representational drift in the mouse visual cortex. *bioRxiv*, 2020.

[10] Tyler D Marks and Michael J Goard. Stimulus-dependent representational drift in primary visual cortex. *bioRxiv*, 2020.

[11] Carl E Schoonover, Sarah N Ohashi, Richard Axel, and Andrew JP Fink. Representational drift in primary olfactory cortex. *bioRxiv*, 2020.

[12] Eve Marder and Jean-Marc Goaillard. Variability, compensation and homeostasis in neuron and network function. *Nature Reviews Neuroscience*, 7(7):563–574, 2006.

[13] Graeme W Davis. Homeostatic control of neural activity: from phenomenology to molecular design. *Annu. Rev. Neurosci.*, 29:307–323, 2006.

[14] Timothy O'Leary and David JA Wyllie. Neuronal homeostasis: time for a change? *The Journal of physiology*, 589(20):4811–4826, 2011.

[15] Graeme W Davis. Homeostatic signaling and the stabilization of neural function. *Neuron*, 80(3):718–728, 2013.

[16] Timothy O'Leary, Alex H Williams, Alessio Franci, and Eve Marder. Cell types, network homeostasis, and pathological compensation from a biologically plausible ion channel expression model. *Neuron*, 82(4):809–821, 2014.

[17] Timothy O'Leary, Alex H Williams, Jonathan S Caplan, and Eve Marder. Correlations in ion channel expression emerge from homeostatic tuning rules. *Proceedings of the National Academy of Sciences*, 110(28):E2645–E2654, 2013.

[18] Michael E Rule, Timothy O'Leary, and Christopher D Harvey. Causes and consequences of representational drift. *Current opinion in neurobiology*, 58:141–147, 2019.

[19] Michael E Rule, Adrianna R Loback, Dhruva V Raman, Laura N Driscoll, Christopher D Harvey, and Timothy O'Leary. Stable task information from an unstable neural population. *Elife*, 9:e51121, 2020.

[20] Niraj S Desai, Lana C Rutherford, and Gina G Turrigiano. Plasticity in the intrinsic excitability of cortical pyramidal neurons. *Nature neuroscience*, 2(6):515–520, 1999.

[21] Timothy O'Leary, Mark CW van Rossum, and David JA Wyllie. Homeostasis of intrinsic excitability in hippocampal neurones: dynamics and mechanism of the response to chronic depolarization. *The Journal of physiology*, 588(1):157–170, 2010.

[22] Corette J Wierenga, Keiji Ibata, and Gina G Turrigiano. Postsynaptic expression of homeostatic plasticity at neocortical synapses. *Journal of Neuroscience*, 25(11): 2895–2905, 2005.

[23] Corette J Wierenga, Michael F Walsh, and Gina G Turrigiano. Temporal regulation of the expression locus of homeostatic plasticity. *Journal of neurophysiology*, 96 (4):2127–2133, 2006.

[24] Gina Turrigiano. Homeostatic synaptic plasticity: local and global mechanisms for stabilizing neuronal function. *Cold Spring Harbor perspectives in biology*, 4(1): a005736, 2012.

[25] Alex H Williams, Timothy O'Leary, and Eve Marder. Homeostatic regulation of neuronal excitability. *Scholarpedia*, 8(1):1656, 2013.

[26] Keith B Hengen, Mary E Lambo, Stephen D Van Hooser, Donald B Katz, and Gina G Turrigiano. Firing rate homeostasis in visual cortex of freely behaving rodents. *Neuron*, 80(2):335–342, 2013.

[27] Carlo NG Giachello and Richard A Baines. Regulation of motoneuron excitability and the setting of homeostatic limits. *Current opinion in neurobiology*, 43:1–6, 2017.

[28] Jonathan Cannon and Paul Miller. Synaptic and intrinsic homeostasis cooperate to optimize single neuron response properties and tune integrator circuits. *Journal of neurophysiology*, 116(5):2004–2022, 2016.

[29] Jonathan Cannon and Paul Miller. Stable control of firing rate mean and variance by dual homeostatic mechanisms. *The Journal of Mathematical Neuroscience*, 7(1): 1–38, 2017.

[30] Paul Miller and Jonathan Cannon. Combined mechanisms of neural firing rate homeostasis. *Biological cybernetics*, 113(1-2):47–59, 2019.

[31] Gina G Turrigiano. The self-tuning neuron: synaptic scaling of excitatory synapses. *Cell*, 135(3):422–435, 2008.

[32] Alanna J Watt and Niraj S Desai. Homeostatic plasticity and stdp: keeping a neuron's cool in a fluctuating world. *Frontiers in synaptic neuroscience*, 2:5, 2010.

[33] Taro Toyoizumi, Megumi Kaneko, Michael P Stryker, and Kenneth D Miller. Modeling the dynamic interaction of hebbian and homeostatic plasticity. *Neuron*, 84 (2):497–510, 2014.

[34] Friedemann Zenke and Wulfram Gerstner. Hebbian plasticity requires compensatory processes on multiple timescales. *Philosophical Transactions of the Royal Society B: Biological Sciences*, 372(1715):20160259, 2017.

[35] Kevin Fox and Michael Stryker. Integrating hebbian and homeostatic plasticity: introduction, 2017.

[36] Gina G Turrigiano. Homeostatic plasticity in neuronal networks: the more things change, the more they stay the same. *Trends in neurosciences*, 22(5):221–227, 1999.

[37] Mark CW Van Rossum, Guo Qiang Bi, and Gina G Turrigiano. Stable hebbian learning from spike timing-dependent plasticity. *Journal of neuroscience*, 20(23): 8812–8821, 2000.

[38] Gina G Turrigiano and Sacha B Nelson. Hebb and homeostasis in neuronal plasticity. *Current opinion in neurobiology*, 10(3):358–364, 2000.

[39] Gina Turrigiano. Too many cooks? intrinsic and synaptic homeostatic mechanisms in cortical circuit refinement. *Annual review of neuroscience*, 34:89–103, 2011.

[40] Daniel Harnack, Miha Pelko, Antoine Chaillet, Yacine Chitour, and Mark CW van Rossum. Stability of neuronal networks with homeostatic regulation. *PLoS computational biology*, 11(7):e1004357, 2015.

[41] Tara Keck, Taro Toyoizumi, Lu Chen, Brent Doiron, Daniel E Feldman, Kevin Fox, Wulfram Gerstner, Philip G Haydon, Mark Hübener, Hey-Kyoung Lee, et al. Integrating hebbian and homeostatic plasticity: the current state of the field and future research directions. *Philosophical Transactions of the Royal Society B: Biological Sciences*, 372(1715):20160158, 2017.

[42] Claudia Clopath, Tobias Bonhoeffer, Mark Hübener, and Tobias Rose. Variance and invariance of neuronal long-term representations. *Philosophical Transactions of the Royal Society B: Biological Sciences*, 372(1715):20160161, 2017.

[43] Zhengyu Ma, Gina G Turrigiano, Ralf Wessel, and Keith B Hengen. Cortical circuit dynamics are homeostatically tuned to criticality in vivo. *Neuron*, 104(4): 655–664, 2019.

[44] Yue Kris Wu, Keith B Hengen, Gina G Turrigiano, and Julijana Gjorgjieva. Homeostatic mechanisms regulate distinct aspects of cortical circuit dynamics. *Proceedings of the National Academy of Sciences*, 117(39):24514–24525, 2020.

[45] Laurence Aitchison and Máté Lengyel. With or without you: predictive coding and bayesian inference in the brain. *Current opinion in neurobiology*, 46:219–227, 2017.

[46] Veronika Koren and Sophie Denève. Computational account of spontaneous activity as a signature of predictive coding. *PLoS computational biology*, 13(1):e1005355, 2017.

[47] James CR Whittington and Rafal Bogacz. An approximation of the error backpropagation algorithm in a predictive coding network with local hebbian synaptic plasticity. *Neural computation*, 29(5):1229–1262, 2017.

[48] Matthew Chalk, Olivier Marre, and Gašper Tkačik. Toward a unified theory of efficient, predictive, and sparse coding. *Proceedings of the National Academy of Sciences*, 115(1):186–191, 2018.

[49] Georg B Keller and Thomas D Mrsic-Flogel. Predictive processing: a canonical cortical computation. *Neuron*, 100(2):424–435, 2018.

[50] Robert Ajemian, Alessandro D'Ausilio, Helene Moorman, and Emilio Bizzi. A theory for how sensorimotor skills are learned and retained in noisy and nonstationary neural circuits. *Proceedings of the National Academy of Sciences*, 110(52):E5078–E5087, 2013.

[51] Alon Rubin, Nitzan Geva, Liron Sheintuch, and Yaniv Ziv. Hippocampal ensemble dynamics timestamp events in long-term memory. *Elife*, 4:e12247, 2015.

[52] Maxwell Gillett, Ulises Pereira, and Nicolas Brunel. Characteristics of sequential activity in networks with temporally asymmetric hebbian learning. *Proceedings of the National Academy of Sciences*, 117(47):29948–29958, 2020.

[53] Dhruva V Raman and Timothy O'leary. Optimal synaptic dynamics for memory maintenance in the presence of noise. *BioRxiv*, 2020.

[54] Daniel Acker, Suzanne Paradis, and Paul Miller. Stable memory and computation in randomly rewiring neural networks. *Journal of neurophysiology*, 2019.

[55] Lee Susman, Naama Brenner, and Omri Barak. Stable memory with unstable synapses. *Nature communications*, 10(1):1–9, 2019.

[56] Michael J Fauth, Florentin Wörgötter, and Christian Tetzlaff. Collective information storage in multiple synapses enables fast learning and slow forgetting. *BMC Neuroscience*, 16(1):1–1, 2015.

[57] Michael Fauth, Florentin Wörgötter, and Christian Tetzlaff. The formation of multi-synaptic connections by the interaction of synaptic and structural plasticity and their functional consequences. *PLoS Comput Biol*, 11(1): e1004031, 2015.

[58] Michael Jan Fauth and Mark CW van Rossum. Self-organized reactivation maintains and reinforces memories despite synaptic turnover. *ELife*, 8:e43717, 2019.

[59] Felipe Yaroslav Kalle Kossio, Sven Goedeke, Christian Klos, and Raoul-Martin Memmesheimer. Drifting assemblies for persistent memory. *bioRxiv*, 2020.

[60] Ashok Litwin-Kumar and Brent Doiron. Formation and maintenance of neuronal assemblies through synaptic plasticity. *Nature communications*, 5(1):1–12, 2014.

[61] William Mau, Michael E Hasselmo, and Denise J Cai. The brain in motion: How ensemble fluidity drives memory-updating and flexibility. *Elife*, 9:e63550, 2020.

[62] Carsen Stringer, Marius Pachitariu, Nicholas Steinmetz, Matteo Carandini, and Kenneth D Harris. High-dimensional geometry of population responses in visual cortex. *Nature*, 571(7765):361–365, 2019.

[63] Zachary P Kilpatrick and Bard Ermentrout. Wandering bumps in stochastic neural fields. *SIAM Journal on Applied Dynamical Systems*, 12(1):61–94, 2013.

[64] Zachary P Kilpatrick, Bard Ermentrout, and Brent Doiron. Optimizing working memory with heterogeneity of recurrent cortical excitation. *Journal of neuroscience*, 33(48):18999–19011, 2013.

[65] Matthew F Panichello, Brian DePasquale, Jonathan W Pillow, and Timothy J Buschman. Error-correcting dynamics in visual working memory. *Nature communications*, 10(1):1–11, 2019.

[66] Jose M Carmena, Mikhail A Lebedev, Craig S Henriquez, and Miguel AL Nicolelis. Stable ensemble performance with single-neuron variability during reaching movements in primates. *Journal of Neuroscience*, 25 (46):10712–10716, 2005.

[67] Byron M Yu, John P Cunningham, Gopal Santhanam, Stephen I Ryu, Krishna V Shenoy, and Maneesh Sahani. Gaussian-process factor analysis for low-dimensional single-trial analysis of neural population activity. *Journal of neurophysiology*, 102(1):614–635, 2009.

[68] Mark M Churchland, John P Cunningham, Matthew T Kaufman, Justin D Foster, Paul Nuyujukian, Stephen I Ryu, and Krishna V Shenoy. Neural population dynamics during reaching. *Nature*, 487(7405):51–56, 2012.

[69] David Sussillo and Omri Barak. Opening the black box: low-dimensional dynamics in high-dimensional recurrent neural networks. *Neural computation*, 25(3):626–649, 2013.

[70] Krishna V Shenoy, Maneesh Sahani, and Mark M Churchland. Cortical control of arm movements: a dynamical systems perspective. *Annual review of neuroscience*, 36:337–359, 2013.

[71] John P Cunningham and Byron M Yu. Dimensionality reduction for large-scale neural recordings. *Nature neuroscience*, 17(11):1500, 2014.

[72] Carlos E Vargas-Irwin, Lachlan Franquemont, Michael J Black, and John P Donoghue. Linking objects to actions: encoding of target object and grasping strategy in primate ventral premotor cortex. *Journal of Neuroscience*, 35(30):10888–10897, 2015.

[73] Mehdi Aghagolzadeh and Wilson Truccolo. Inference and decoding of motor cortex low-dimensional dynamics via latent state-space models. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, 24(2): 272–282, 2015.

[74] Wilson Truccolo. From point process observations to collective neural dynamics: Nonlinear hawkes process glms, low-dimensional dynamics and coarse graining. *Journal of Physiology-Paris*, 110(4):336–347, 2016.

[75] Juan A Gallego, Matthew G Perich, Lee E Miller, and Sara A Solla. Neural manifolds for the control of movement. *Neuron*, 94(5):978–984, 2017.

[76] Juan A. Gallego, Matthew G. Perich, Stephanie N. Naufel, Christian Ethier, Sara A. Solla, and Lee E. Miller. Cortical population activity within a preserved neural manifold underlies multiple motor behaviors. *Nature Communications*, 9(1):4233, dec 2018. ISSN 2041-1723. doi: 10.1038/s41467-018-06560-z.

[77] Alex H Williams, Tony Hyun Kim, Forea Wang, Saurabh Vyas, Stephen I Ryu, Krishna V Shenoy, Mark Schnitzer, Tamara G Kolda, and Surya Ganguli. Unsupervised discovery of demixed, low-dimensional neural dynamics across multiple timescales through tensor component analysis. *Neuron*, 98(6):1099–1115, 2018.

[78] Stefano Recanatesi, Matthew Farrell, Guillaume Lajoie, Sophie Deneve, Mattia Rigotti, and Eric Shea-Brown. Signatures and mechanisms of low-dimensional neural predictive manifolds. *bioRxiv*, page 471987, 2018.

[79] Carsen Stringer, Marius Pachitariu, Nicholas Steinmetz, Charu Bai Reddy, Matteo Carandini, and Kenneth D Harris. Spontaneous behaviors drive multidimensional, brainwide activity. *Science*, 364(6437):255–255, 2019.

[80] David Xing, Mehdi Aghagolzadeh, Wilson Truccolo, and David Borton. Low-dimensional motor cortex dynamics preserve kinematics information during unconstrained locomotion in nonhuman primates. *Frontiers in neuroscience*, 13:1046, 2019.

[81] Alon Rubin, Liron Sheintuch, Noa Brande-Eilat, Or Pinchasof, Yoav Rechavi, Nitzan Geva, and Yaniv Ziv. Revealing neural correlates of behavior without behavioral measurements. *Nature Communications*, 10 (1):4745, dec 2019. ISSN 2041-1723. doi: 10.1038/s41467-019-12724-2.

[82] Hamidreza Abbaspourazad, Mahdi Choudhury, Yan T Wong, Bijan Pesaran, and Maryam M Shanechi. Multiscale low-dimensional motor cortical state dynamics predict naturalistic reach-and-grasp behavior. *Nature communications*, 12(1):1–19, 2021.

[83] Juan A Gallego, Matthew G Perich, Raeed H Chowdhury, Sara A Solla, and Lee E Miller. Long-term stability of cortical population dynamics underlying consistent behavior. *Nature neuroscience*, 23(2):260–270, 2020.

[84] Eva L. Dyer, Mohammad Gheshlaghi Azar, Matthew G. Perich, Hugo L. Fernandes, Stephanie Naufel, Lee E. Miller, and Konrad P. Körding. A cryptography-based approach for movement decoding. *Nature Biomedical Engineering*, 1(12):967–976, dec 2017. ISSN 2157-846X. doi: 10.1038/s41551-017-0169-7.

[85] Jonathan C. Kao, Stephen I. Ryu, and Krishna V. Shenoy. Leveraging neural dynamics to extend functional lifetime of brain-machine interfaces. *Scientific Reports*, 7 (1):7395, dec 2017. ISSN 2045-2322. doi: 10.1038/s41598-017-06029-x.

[86] Ali Farshchian, Juan A Gallego, Joseph P Cohen, Yoshua Bengio, Lee E Miller, and Sara A Solla. Adversarial domain adaptation for stable brain-machine interfaces. *arXiv preprint arXiv:1810.00045*, 2018.

[87] Alan D. Degenhart, William E. Bishop, Emily R. Oby, Elizabeth C. Tyler-Kabara, Steven M. Chase, Aaron P. Batista, and Byron M. Yu. Stabilization of a brain–computer interface via the alignment of low-dimensional spaces of neural activity. *Nature Biomedical Engineering*, apr 2020. ISSN 2157-846X. doi: 10.1038/s41551-020-0542-9.

[88] Ethan Sorrell, Michael E Rule, and Timothy O'Leary. Brain–machine interfaces: Closed-loop control in an adaptive system. *Annual Review of Control, Robotics, and Autonomous Systems*, 4, 2021.

[89] N Alex Cayco-Gajic, Claudia Clopath, and R Angus Silver. Sparse synaptic connectivity is required for decorrelation and pattern separation in feedforward networks. *Nature communications*, 8(1):1–11, 2017.

[90] Ashok Litwin-Kumar, Kameron Decker Harris, Richard Axel, Haim Sompolinsky, and LF Abbott. Optimal degrees of synaptic connectivity. *Neuron*, 93(5):1153–1164, 2017.

[91] Eve Marder and Lamont S Tang. Coordinating different homeostatic processes. *Neuron*, 66(2):161–163, 2010.

[92] Matteo Carandini and David J Heeger. Normalization as a canonical neural computation. *Nature Reviews Neuroscience*, 13(1):51–62, 2012.

[93] Lingxuan Chen, Kirstie A Cummings, William Mau, Yosif Zaki, Zhe Dong, Sima Rabinowitz, Roger L Clem, Tristan Shuman, and Denise J Cai. The role of intrinsic excitability in the evolution of memory: Significance in memory allocation, consolidation, and updating. *Neurobiology of Learning and Memory*, 173:107266, 2020.

[94] Pietro Berkes, Gergő Orbán, Máté Lengyel, and József Fiser. Spontaneous cortical activity reveals hallmarks of an optimal internal model of the environment. *Science*, 331(6013):83–87, 2011.

[95] Meytal Wilf, Francesca Strappini, Tal Golan, Avital Hahamy, Michal Harel, and Rafael Malach. Spontaneously emerging patterns in human visual cortex reflect responses to naturalistic sensory stimuli. *Cerebral cortex*, 27(1):750–763, 2017.

[96] Ralph Bourdoukan, David Barrett, Sophie Deneve, and Christian K Machens. Learning optimal spike-based representations. *Advances in neural information processing systems*, 25:2285–2293, 2012.

[97] Sophie Denève and Christian K Machens. Efficient codes and balanced networks. *Nature neuroscience*, 19 (3):375–382, 2016.

[98] Sophie Denève, Alireza Alemi, and Ralph Bourdoukan. The brain as an efficient and robust adaptive learner. *Neuron*, 94(5):969–977, 2017.

[99] Cristina Savin and Sophie Deneve. Spatio-temporal representations of uncertainty in spiking neural networks. In *NIPS*, volume 27, pages 2024–2032, 2014.

[100] Wieland Brendel, Ralph Bourdoukan, Pietro Vertechi, Christian K Machens, and Sophie Denéve. Learning to represent signals spike by spike. *PLoS computational biology*, 16(3):e1007692, 2020.

[101] JHM Wong and MJF Gales. Sequence student-teacher training of deep neural networks. In *Proceedings of the Annual Conference of the International Speech Communication Association, INTERSPEECH*, volume 8, pages 2761–2765, 2016.

[102] Linfeng Zhang, Jiebo Song, Anni Gao, Jingwei Chen, Chenglong Bao, and Kaisheng Ma. Be your own teacher: Improve the performance of convolutional neural networks via self distillation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3713–3722, 2019.

[103] Erkki Oja. Simplified neuron model as a principal component analyzer. *Journal of mathematical biology*, 15(3): 267–273, 1982.

[104] Kenneth D Miller and David JC MacKay. The role of constraints in hebbian learning. *Neural computation*, 6 (1):100–126, 1994.

[105] Anirvan Sengupta, Cengiz Pehlevan, Mariano Tepper, Alexander Genkin, and Dmitri Chklovskii. Manifold-tiling localized receptive fields are optimal in similarity-preserving neural networks. In *Advances in Neural Information Processing Systems*, pages 7080–7090, 2018.

[106] Alexander Genkin, Anirvan M Sengupta, and Dmitri Chklovskii. A neural network for semi-supervised learning on manifolds. In *International Conference on Artificial Neural Networks*, pages 375–386. Springer, 2019.

[107] Dmitry Krotov and John J Hopfield. Unsupervised learning by competing hidden units. *Proceedings of the National Academy of Sciences*, 116(16):7723–7731, 2019.

[108] Cengiz Pehlevan, Sreyas Mohan, and Dmitri B Chklovskii. Blind nonnegative source separation using biological neural networks. *Neural computation*, 29(11): 2925–2954, 2017.

[109] Cengiz Pehlevan, Anirvan M Sengupta, and Dmitri B Chklovskii. Why do similarity matching objectives lead to hebbian/anti-hebbian networks? *Neural computation*, 30(1):84–124, 2018.

[110] Cengiz Pehlevan, Xinyuan Zhao, Anirvan M Sengupta, and Dmitri Chklovskii. Neurons as canonical correlation analyzers. *Frontiers in computational neuroscience*, 14:55, 2020.

[111] David Lipshutz, Yanis Bahroun, Siavash Golkar, Anirvan M Sengupta, and Dmitri B Chkovskii. A biologically plausible neural network for multi-channel canonical correlation analysis. *arXiv preprint arXiv:2010.00525*, 2020.

[112] Stefano Recanatesi, Matthew Farrell, Guillaume Lajoie, Sophie Deneve, Mattia Rigotti, and Eric Shea-Brown. Predictive learning extracts latent space representations from sensory observations. *bioRxiv*, page 471987, 2019.

[113] Kalman A Katlowitz, Michel A Picardo, and Michael A Long. Stable sequential activity underlying the maintenance of a precisely executed skilled behavior. *Neuron*, 98(6):1133–1140, 2018.

[114] Thomas Hainmueller and Marlene Bartos. Parallel emergence of stable and dynamic memory engrams in the hippocampus. *Nature*, 558(7709):292–296, 2018.

[115] Yann Sweeney and Claudia Clopath. Population coupling predicts the plasticity of stimulus responses in cortical circuits. *bioRxiv*, page 265041, 2018.

[116] Uri Rokni, Andrew G Richardson, Emilio Bizzi, and H Sebastian Seung. Motor learning with unstable neural representations. *Neuron*, 54(4):653–666, 2007.

[117] Daniel Huber, Diego A Gutnisky, Simon Peron, Daniel H O'connor, J Simon Wiegert, Lin Tian, Thomas G Oertner, Loren L Looger, and Karel Svoboda. Multiple dynamic representations in the motor cortex during sensorimotor learning. *Nature*, 484(7395):473, 2012.

[118] Ashesh K Dhawale, Maurice A Smith, and Bence P Ölveczky. The role of variability in motor learning. *Annual review of neuroscience*, 40:479–498, 2017.

[119] Alessio Attardo, Ju Lu, Takashi Kawashima, Hiroyuki Okuno, James E Fitzgerald, Haruhiko Bito, and Mark J Schnitzer. Long-term consolidation of ensemble neural plasticity patterns in hippocampal area CA1. *Cell reports*, 25(3):640–650, 2018.

[120] Albert You, Ellen L Zippi, and Jose M Carmena. Large-scale neural consolidation in bmi learning. In *2019 9th International IEEE/EMBS Conference on Neural Engineering (NER)*, pages 603–606. IEEE, 2019.

[121] Albert K You, Bing Liu, Abhimanyu Singhal, Suraj Gowda, Helene Moorman, Amy Orsborn, Karunesh Ganguly, and Jose M Carmena. Flexible modulation of neural variance facilitates neuroprosthetic skill learning. *BioRxiv*, page 817346, 2019.

[122] Karunesh Ganguly and Jose M. Carmena. Neural correlates of skill acquisition with a cortical brain–machine interface. *Journal of Motor Behavior*, 42(6):355–360, oct 2010. ISSN 0022-2895, 1940-1027. doi: 10.1080/00222895.2010.526457.

[123] Robert M French. Catastrophic forgetting in connectionist networks. *Trends in cognitive sciences*, 3(4):128–135, 1999.

[124] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, page 201611835, 2017.

[125] Friedemann Zenke, Ben Poole, and Surya Ganguli. Continual learning through synaptic intelligence. In *International Conference on Machine Learning*, pages 3987–3995, 2017.

[126] Nicolas Y. Masse, Gregory D. Grant, and David J. Freedman. Alleviating catastrophic forgetting using context-dependent gating and synaptic stabilization. *Proceedings of the National Academy of Sciences*, 115(44):E10467–E10475, 2018. ISSN 0027-8424. doi: 10.1073/pnas.1803839115.

[127] Lea Duncker, Laura Driscoll, Krishna V Shenoy, Maneesh Sahani, and David Sussillo. Organizing recurrent network dynamics by task-computation to enable continual learning. *Advances in Neural Information Processing Systems*, 33, 2020.

[128] Anthony Robins. Catastrophic forgetting, rehearsal and pseudorehearsal. *Connection Science*, 7(2):123–146, 1995.

[129] Szabolcs Káli and Peter Dayan. Replay, repair and consolidation. *Advances in Neural Information Processing Systems*, pages 19–26, 2003.

[130] Szabolcs Káli and Peter Dayan. Off-line replay maintains declarative memories in a model of hippocampal-neocortical interactions. *Nature neuroscience*, 7(3):286, 2004.

[131] Oscar C González, Yury Sokolov, Giri P Krishnan, Jean Erik Delanois, and Maxim Bazhenov. Can sleep protect memories from catastrophic forgetting? *Elife*, 9:e51005, 2020.

[132] Ronald W Oppenheim. Cell death during development of the nervous system. *Annual review of neuroscience*, 14(1):453–501, 1991.

[133] Javier Iglesias and Alessandro EP Villa. Effect of stimulus-driven pruning on the detection of spatiotemporal patterns of activity in large neural networks. *BioSystems*, 89(1-3):287–293, 2007.

[134] Ed Bullmore and Olaf Sporns. The economy of brain network organization. *Nature Reviews Neuroscience*, 13(5):336–349, 2012.

[135] Elliot J Crowley, Jack Turner, Amos Storkey, and Michael O'Boyle. Pruning neural networks: is it time to nip it in the bud? 2018.

[136] Elliot J Crowley, Jack Turner, Amos Storkey, and Michael O'Boyle. A closer look at structured pruning for neural network compression. *arXiv preprint arXiv:1810.04622*, 2018.

[137] Carolin Scholl, Michael E Rule, and Matthias H Hennig. The information theory of developmental pruning: Optimizing global network architecture using local synaptic rules. *bioRxiv*, 2020.

[138] Gal Chechik, Isaac Meilijson, and Eytan Ruppin. Synaptic pruning in development: a computational account. *Neural computation*, 10(7):1759–1777, 1998.

[139] Dhruva V Raman, Adriana P Rotondo, and Timothy O'Leary. Fundamental bounds on learning performance in neural circuits. *Proceedings of the National Academy of Sciences*, 116(21):10537–10546, 2019.

[140] Julia Steinberg, Madhu Advani, and Haim Sompolinsky. New role for circuit expansion for learning in neural networks. *Physical Review E*, 103(2):022404, 2021.

[141] Matthew D Golub, Patrick T Sadtler, Emily R Oby, Kristin M Quick, Stephen I Ryu, Elizabeth C Tyler-Kabara, Aaron P Batista, Steven M Chase, and M Yu Byron. Learning by neural reassociation. *Nature neuroscience*, 21(4):607–616, 2018.

[142] Emily R Oby, Matthew D Golub, Jay A Hennig, Alan D Degenhart, Elizabeth C Tyler-Kabara, M Yu Byron, Steven M Chase, and Aaron P Batista. New neural activity patterns emerge with long-term learning. *Proceedings of the National Academy of Sciences*, 116(30):15210–15215, 2019.

[143] Barbara Feulner and Claudia Clopath. Neural manifold under plasticity in a goal driven learning behaviour. *bioRχiv*, feb 2020. doi: 10.1101/2020.02.21.959163.

[144] Laura N Driscoll, Noah L Pettit, Matthias Minderer, Selmaan N Chettih, and Christopher D Harvey. Data from: Dynamic reorganization of neuronal activity patterns in parietal cortex dataset. *Dryad*, (Dataset) https://doi.org/10.5061/dryad.gqnk98sjq, 2020.

[145] Bernard Widrow and Marcian E Hoff. Adaptive switching circuits. Technical report, Stanford Univ Ca Stanford Electronics Labs, 1960.

[146] Bernard Widrow and Marcian E Hoff. Associative storage and retrieval of digital information in networks of adaptive "neurons". In *Biological Prototypes and Synthetic Systems*, pages 160–160. Springer, 1962.

[147] Martin Boerlin and Sophie Denève. Spike-based population coding and working memory. *PLoS Comput Biol*, 7(2):e1001080, 2011.