Self Healing Codes

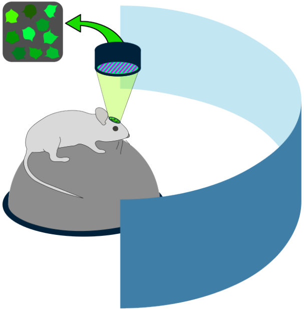
5 November 2020

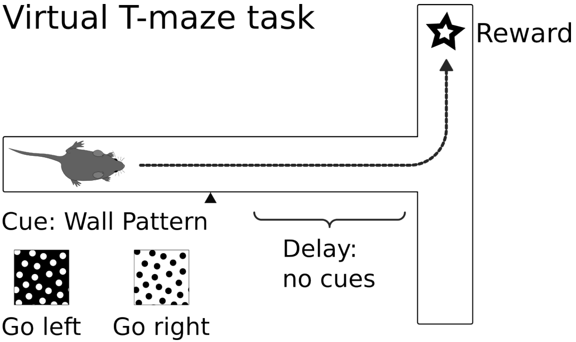
Michael Rule

The brain is plastic.

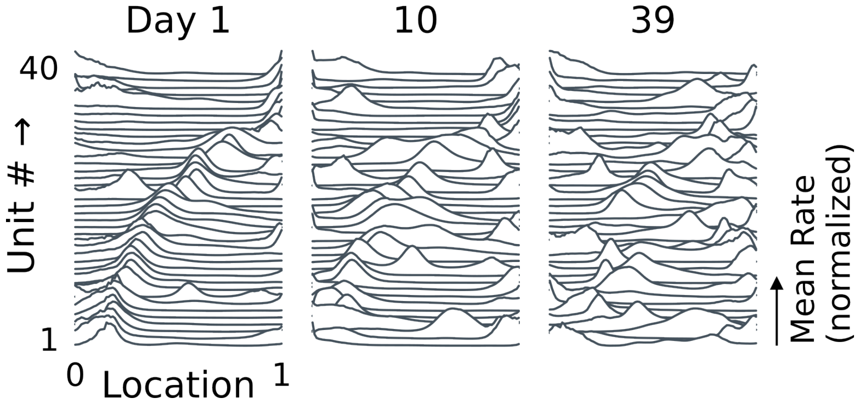
The brain is stable.

Image neural population codes over time





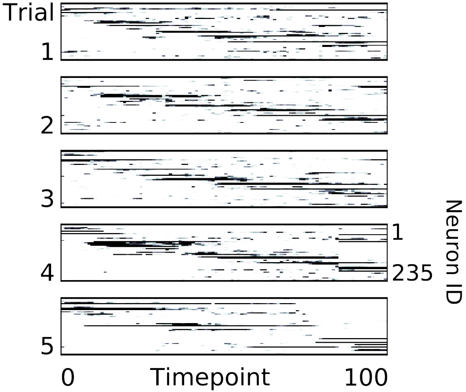
Neural population code is unstable*



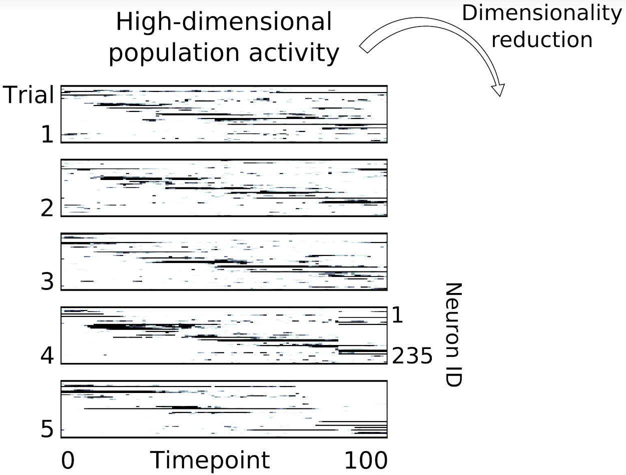
*In Posterior Parietal Cortex (PPC)

Representational Drift

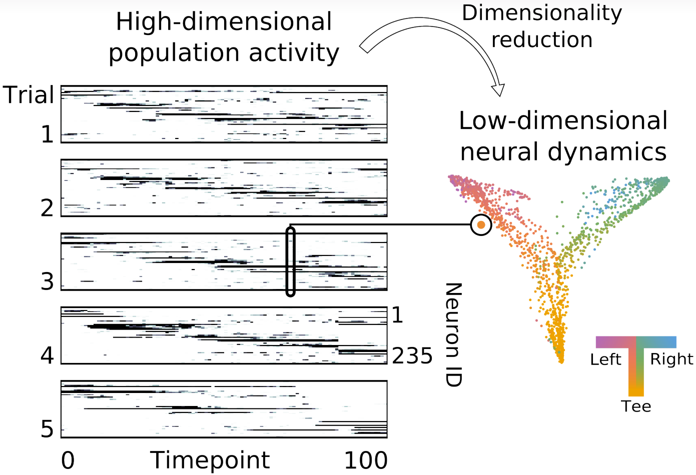
High-dimensional
population activity



Rule, O'Leary, Harvey,(2019) *Causes and consequences of representational drift.*



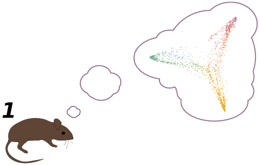
Rule, O'Leary, Harvey,(2019) *Causes and consequences of representational drift.*



Rule, O'Leary, Harvey,(2019) *Causes and consequences of representational drift.*

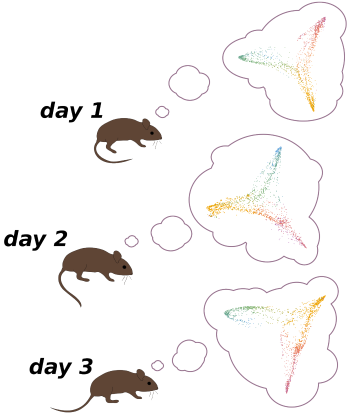
Many degrees of freedom
in internal representations

day 1

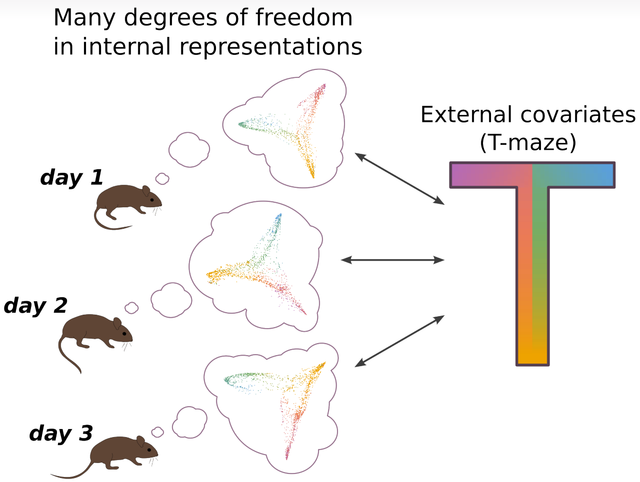


Rule, O'Leary, Harvey,(2019) *Causes and consequences of representational drift.*

Many degrees of freedom
in internal representations



Rule, O'Leary, Harvey,(2019) *Causes and consequences of representational drift.*



Rule, O'Leary, Harvey,(2019) *Causes and consequences of representational drift.*

Stable Task Information

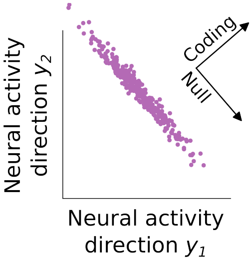
Rule ME, Loback AR, Raman DV, Driscoll L, Harvey CD, O'Leary T. 2020. Stable task information from an unstable neural population. eLife.

How is the brain is robust to changing (Δ) neural codes?

How is the brain is robust to changing (Δ) neural codes?

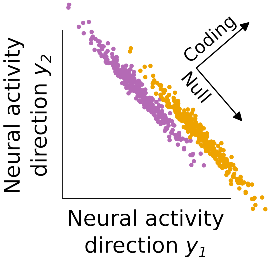
Invariance:

- Δ in null-space of readout



Invariance:

- Δ in null-space of readout



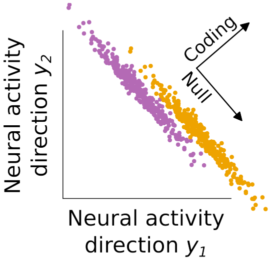
How is the brain is robust to changing (Δ) neural codes?

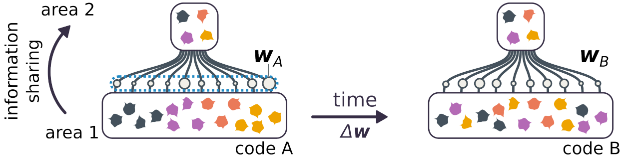
Invariance:

- Δ in null-space of readout

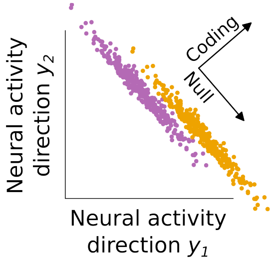
Coordination:

- Slow Δ, readout adapts





How is the brain is robust to changing (Δ) neural codes?



Invariance:

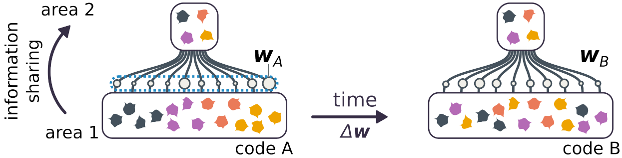
- Δ in null-space of readout

Coordination:

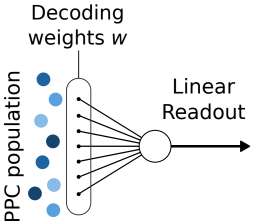
- Slow Δ, readout adapts

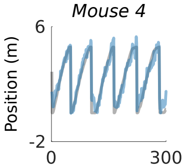
Analyse Driscoll et al. '17

- Δ preserves invariant readout
- Slow plasticity can track Δ

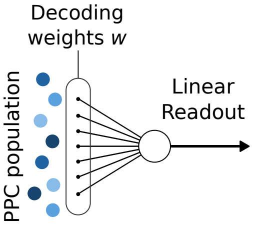


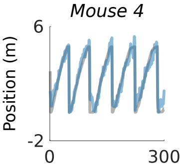
Single-day decoders generalize poorly

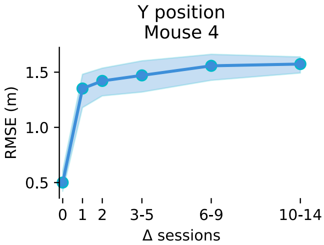




Single-day decoders generalize poorly



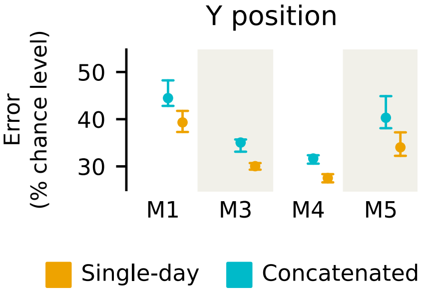




. . . but hint at long-term stable structure

Long-term ≈ stable subspace exists, drift is constrained

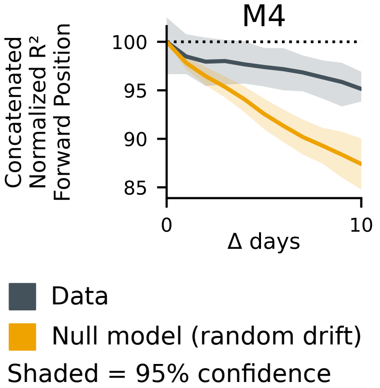
Fixed decoder trained over data from 7-10 days nearly as good as single-day



Long-term ≈ stable subspace exists, drift is constrained

Fixed decoder trained over data from 7-10 days nearly as good as single-day

Unconstrained drift rapidly degrades performance

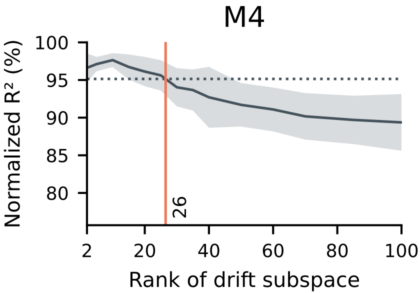


Long-term ≈ stable subspace exists, drift is constrained

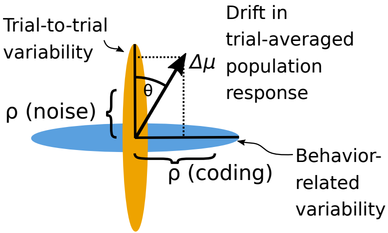
Fixed decoder trained over data from 7-10 days nearly as good as single-day

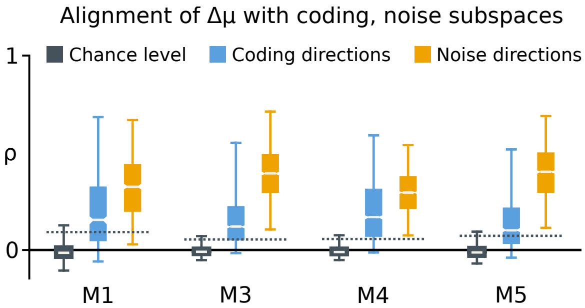
Unconstrained drift rapidly degrades performance

Results consistent with low-rank drift



Drift resembles trial-to-trial variability

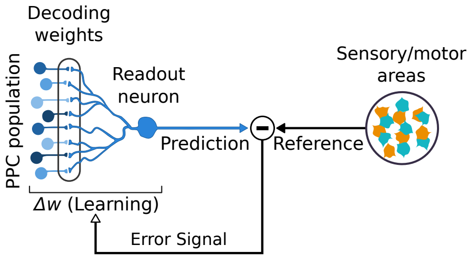


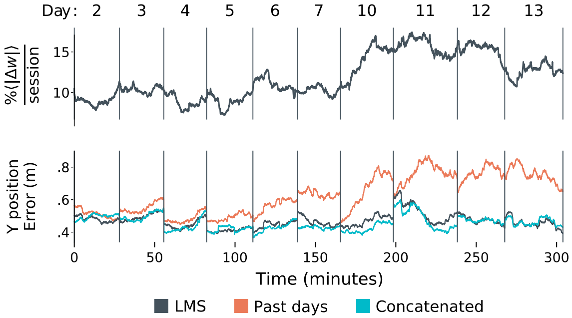


. . . But some drift occurs in directions that encode task information

≈ Stable subspace can be identified, tracked with modest plasticity

Distributed representations could detect tuning changes, adjust decoding weights





(~10-15% weight change per session for ~100 cells, more cells → less plasticity)

Drift (Δ) is structured

- Long-term Δ less than expected
- Consistent with low-rank
- More Δ in null directions

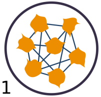
Track $\approx$ stable subspace

- Non-null Δ: slow and easy to track
- Weak error feedback sufficient

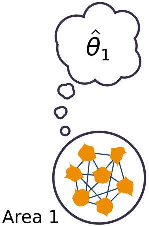
This talk: **Could neurons use what is stable to track what is volatile?**

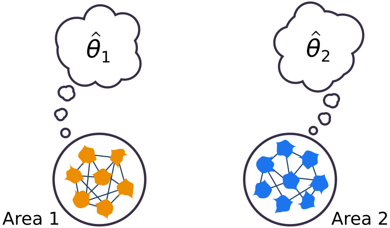
- Learning and correlated activation is sufficient to track drift
- How to stabilize readout without external feedback

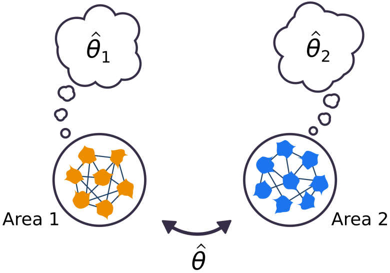
Ongoing learning addresses drift

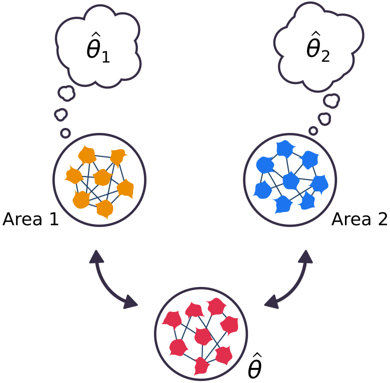


Area 1

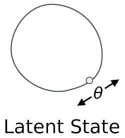

$\hat{\theta}_1$
Area 1

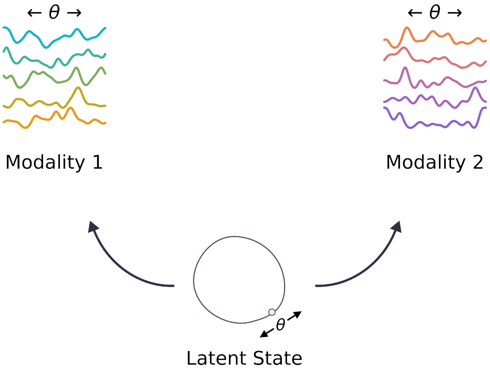

$\hat{\theta}_1$
$\hat{\theta}_2$
Area 1
Area 2


$\hat{\theta}_1$
$\hat{\theta}_2$
Area 1
Area 2
$\hat{\theta}$

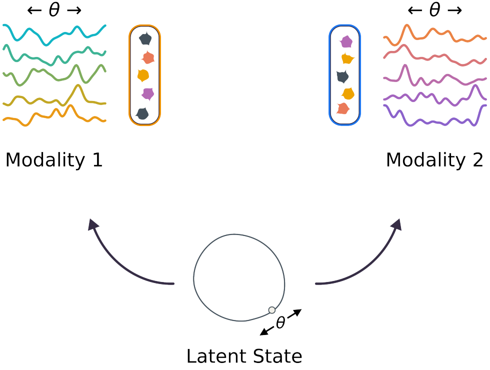

$\hat{\theta}_1$
$\hat{\theta}_2$
Area 1
Area 2
$\hat{\theta}$

- Low-D latent variable



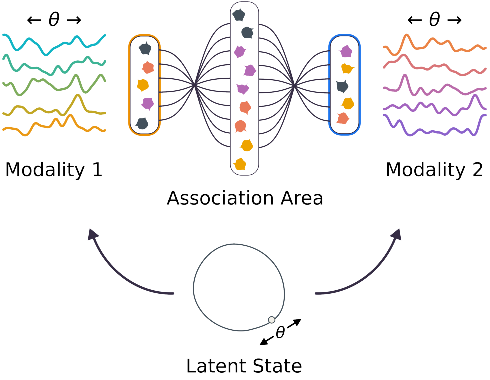


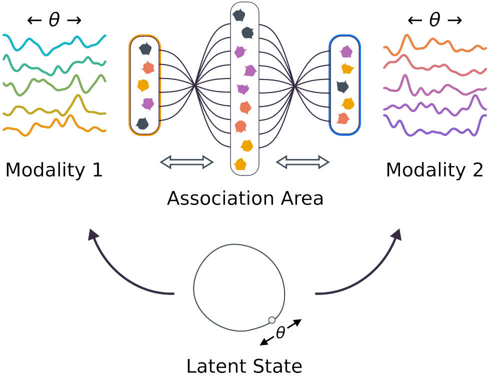
- Low-D latent variable
- Different areas, correlated variability



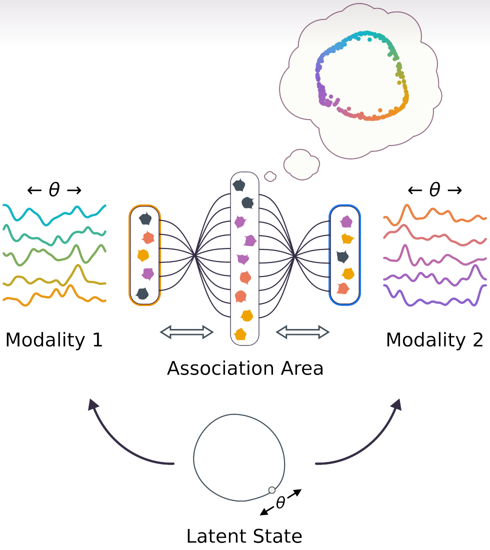
- Low-D latent variable
- Different areas, correlated variability

- Low-D latent variable
- Different areas, correlated variability

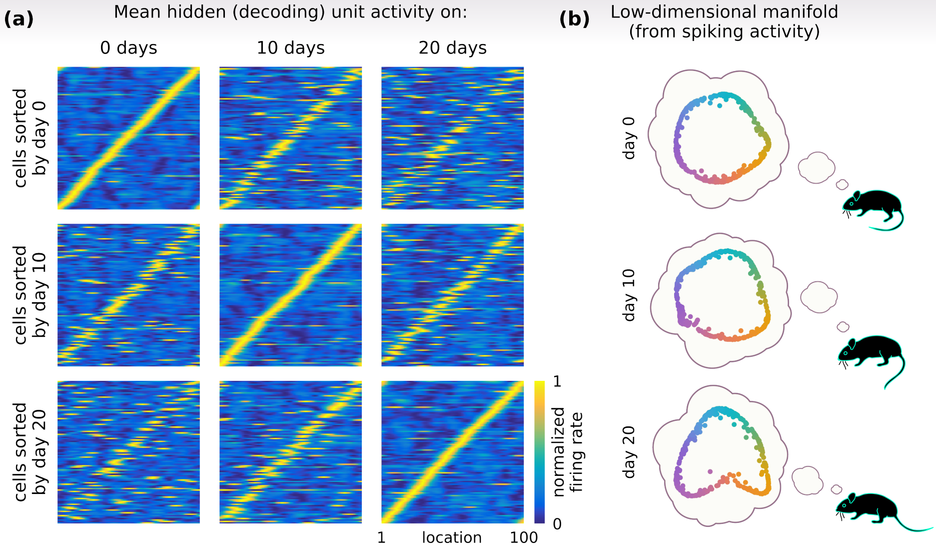




- Low-D latent variable
- Different areas, correlated variability
- Restricted Boltzmann Machine
  - Stochastic, binary
  - Non-negative activity

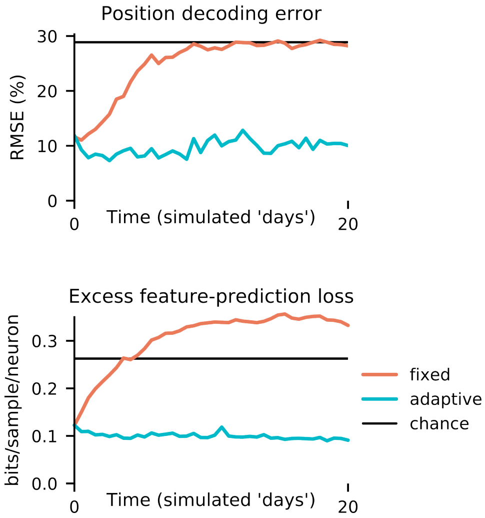


- Low-D latent variable
- Different areas, correlated variability
- Restricted Boltzmann Machine
  - Stochastic, binary
  - Non-negative activity
- Drift: noise to synaptic weights
  - Train continuously
  - Maintain mean rates
  - Normalize population responses (units compete)



(a)
Mean hidden (decoding) unit activity on:
0 days
10 days
20 days
cells sorted by day 0
cells sorted by day 10
cells sorted by day 20
1
location
100
1
0
normalized firing rate
(b)
Low-dimensional manifold (from spiking activity)
day 0
day 10
day 20

Ongoing learning addresses drift



Looks like drift

- Code in "association" area changes
- Population low-D task structure stable

Consistent readout via unsupervised learning

- Correlated activation → error feedback
- Plasticity: readouts to learn as quickly as the representation changes

Problem: Forgets easily

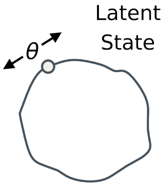
How could stable internal codes coexist with such untsable representations?

Q: How to achieve stable interpretations of unstable codes?

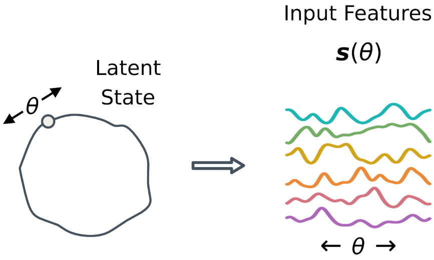
A: Homeostatic mechanisms create stability without error feedback.

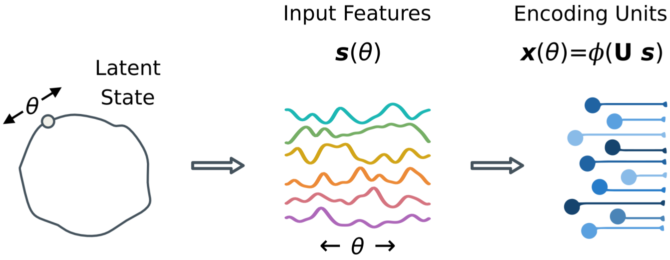
Model Drift

Model encoding drift

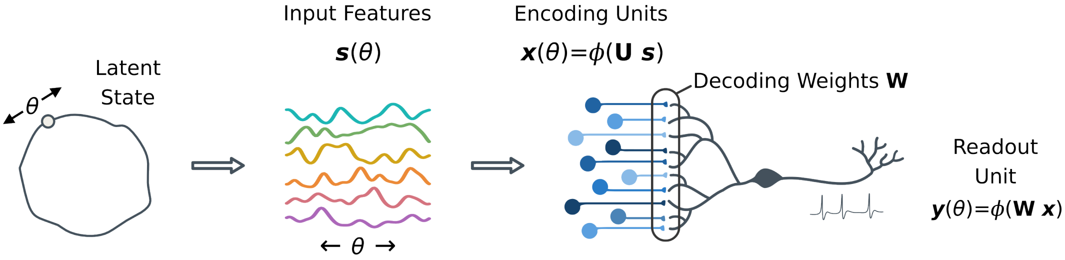


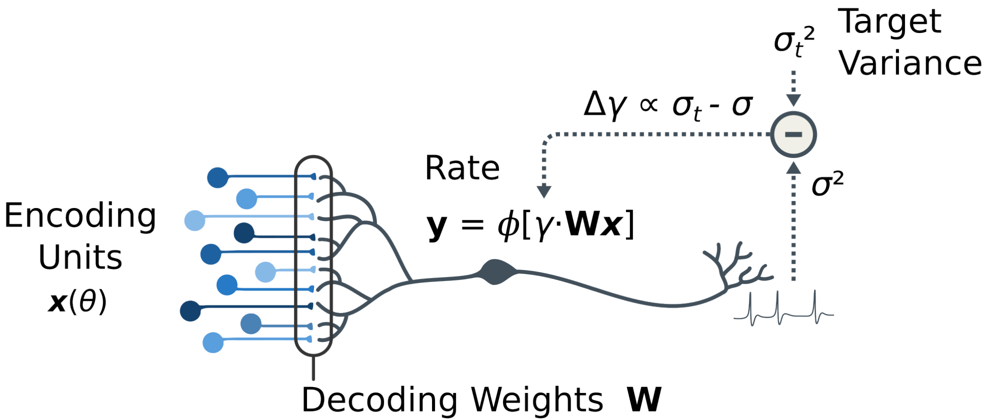
Model encoding drift




Latent
State
θ
Input Features
$\boldsymbol{s}(\theta)$
$\leftarrow \theta \rightarrow$
Encoding Units
$\boldsymbol{x}(\theta)=\phi(\mathbf{U}\ \boldsymbol{s})$

Model encoding drift

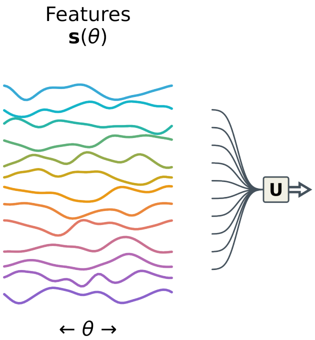




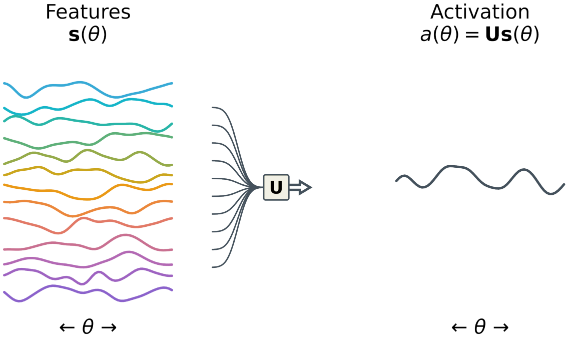
Target Variance
σ_t^2
$\Delta\gamma \propto \sigma_t - \sigma$
Rate
$\mathbf{y} = \phi[\gamma \cdot \mathbf{W}\boldsymbol{x}]$
σ^2
Encoding Units
$\boldsymbol{x}(\theta)$
Decoding Weights $\mathbf{W}$

Neurons extract maxima of activation function over θ

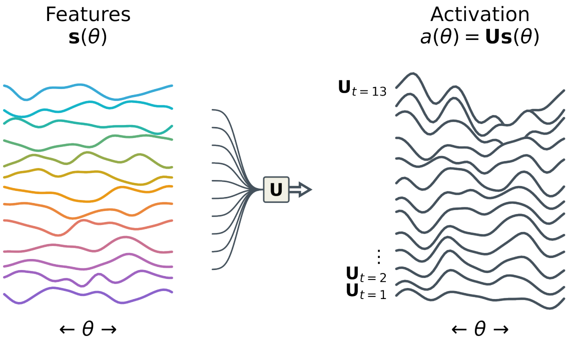
Neurons extract maxima of activation function over θ



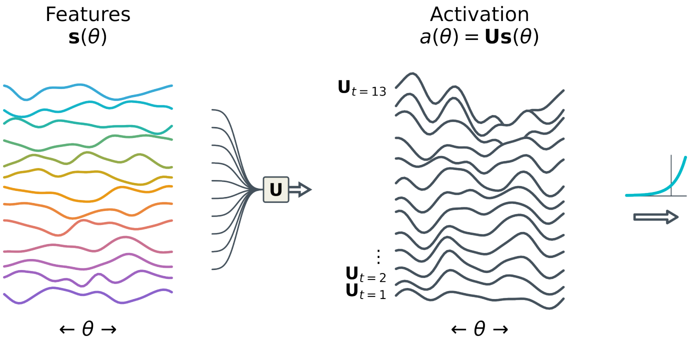
Neurons extract maxima of activation function over θ



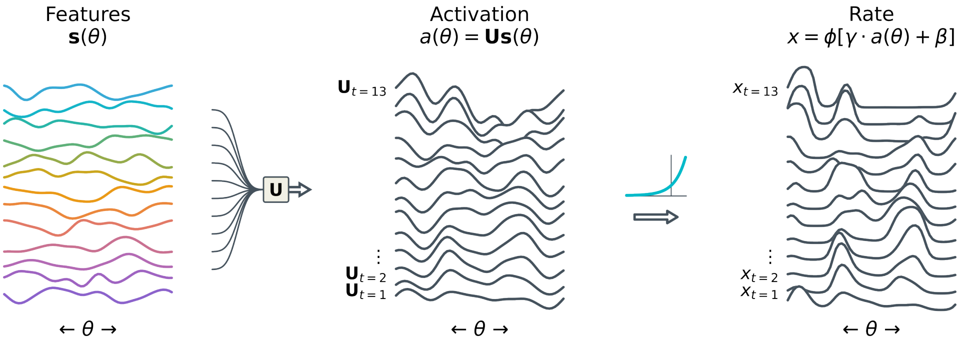
Neurons extract maxima of activation function over θ



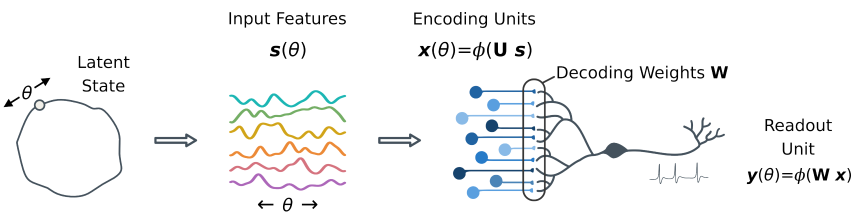
Neurons extract maxima of activation function over θ



Neurons extract maxima of activation function over θ



Hard to change preferred tuning



$$x(\theta) = \mathbf{Us}(\theta), \qquad \theta_0 = \underset{\theta}{\operatorname{argmax}}[x(\theta)]$$

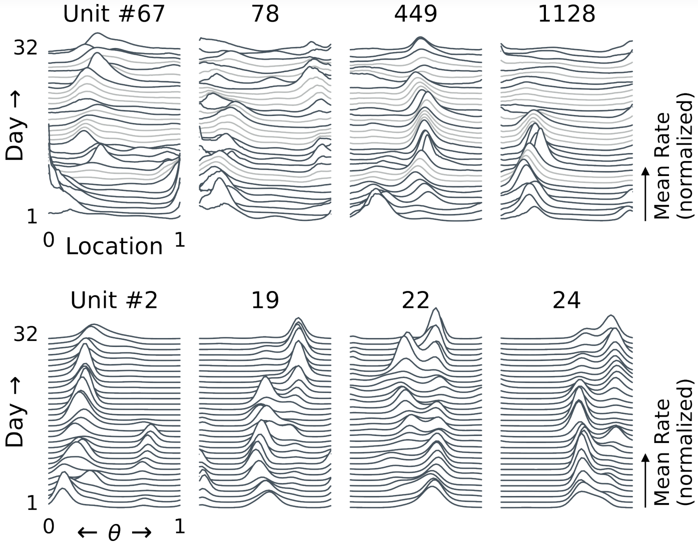
$$\Delta\theta_0 = -\left[\nabla_\theta \nabla_\theta^\top x(\theta)\right]^{-1} [\mathbf{U}\nabla_\theta \Delta\mathbf{s}(\theta)]$$

$\Delta\mathbf{s}(\theta)$ must resemble $\Delta\theta$ near the peak θ_0, only $\nabla_\theta \Delta\mathbf{s}(\theta)$ matters

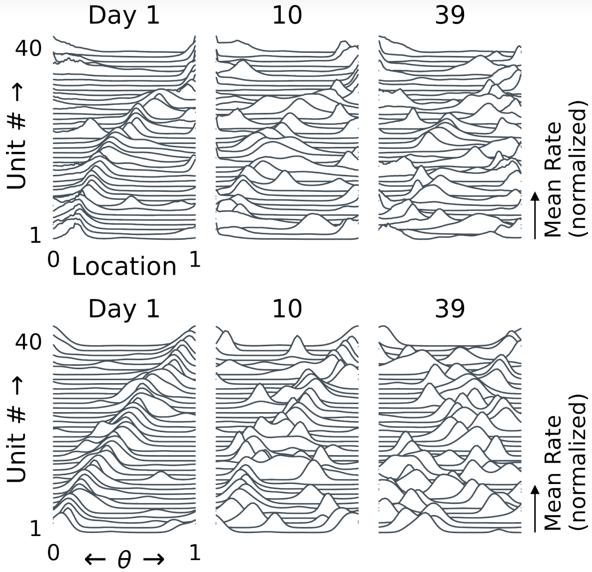
Any $\Delta\mathbf{s}(\theta)$ in null space of $\mathbf{u}$ irrelevant

$\left[\nabla_\theta \nabla_\theta^\top x(\theta)\right]^{-1}$: sharper peaks are harder to move

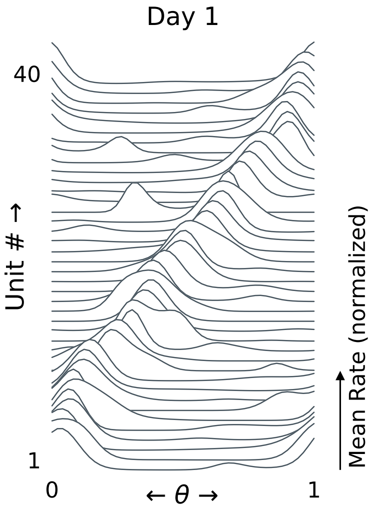
Model encoding drift



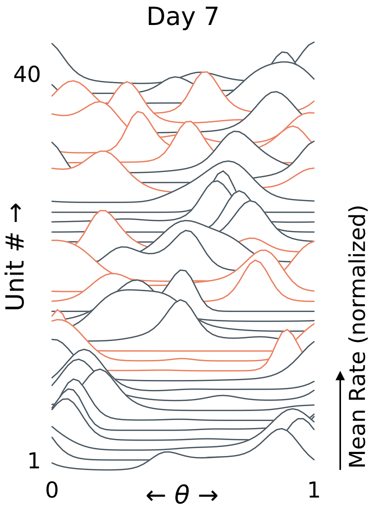
Model encoding drift



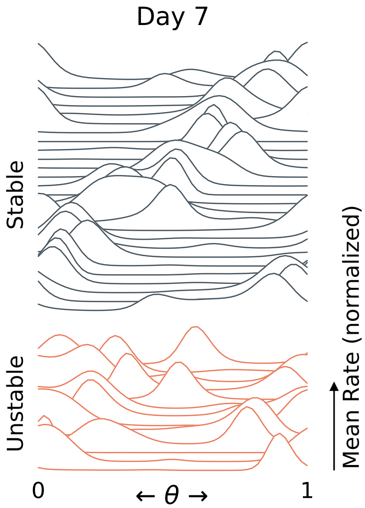
Drift is gradual



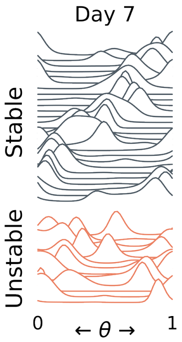
Drift is gradual



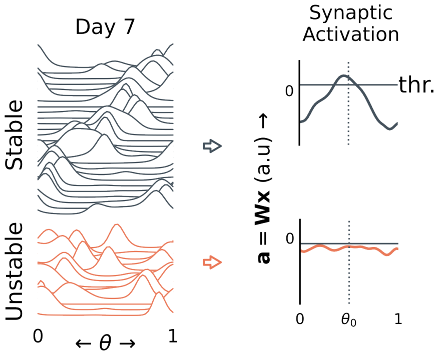
Drift is gradual



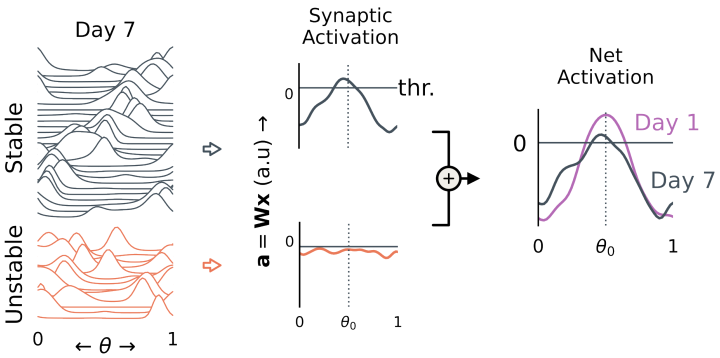
Drift causes loss of excitability, not tuning



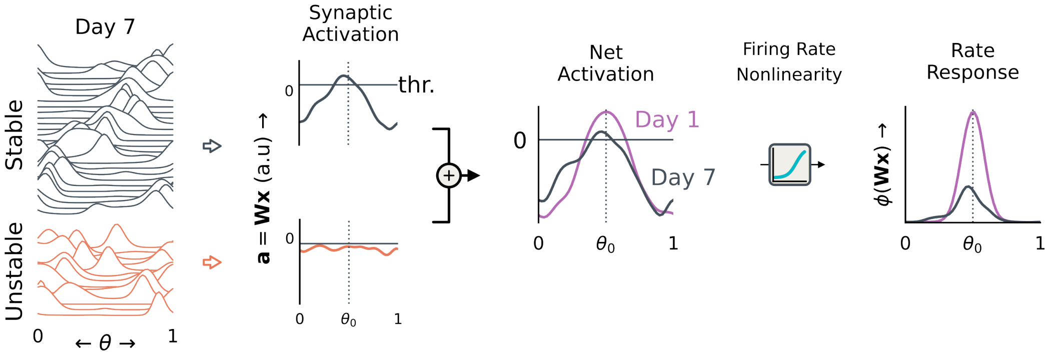
Drift causes loss of excitability, not tuning

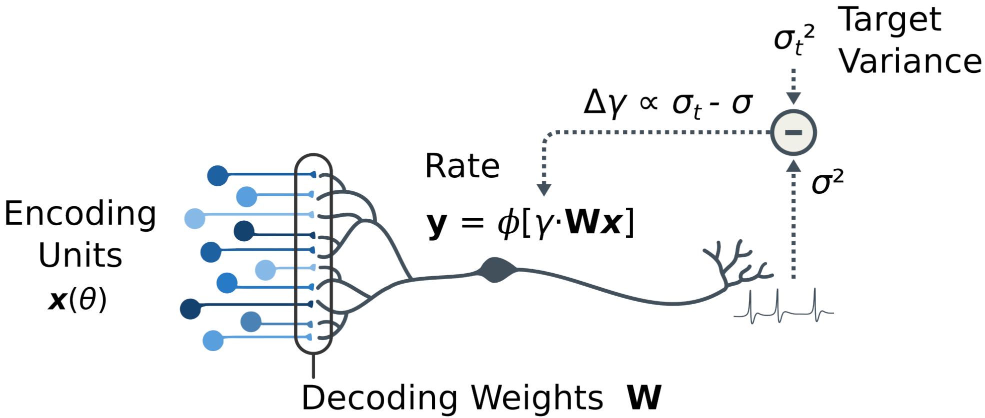


Drift causes loss of excitability, not tuning

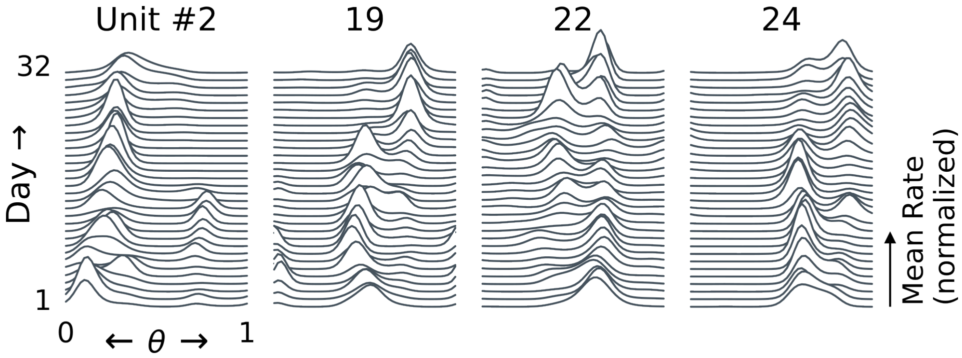


Drift causes loss of excitability, not tuning

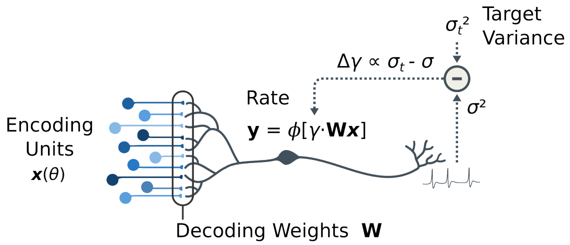



Target Variance
σ_t^2
$\Delta\gamma \propto \sigma_t - \sigma$
Rate
$\mathbf{y} = \phi[\gamma \cdot \mathbf{W}\boldsymbol{x}]$
σ^2
Encoding Units
$\boldsymbol{x}(\theta)$
Decoding Weights $\mathbf{W}$

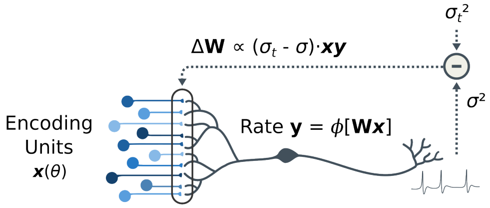
Sensitivity Homeostasis

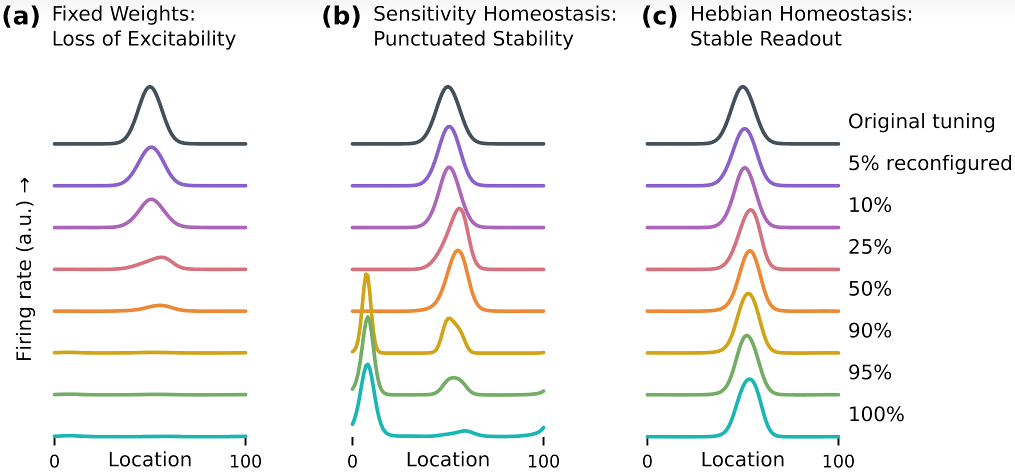


Sensitivity Homeostasis



Hebbian Homeostasis



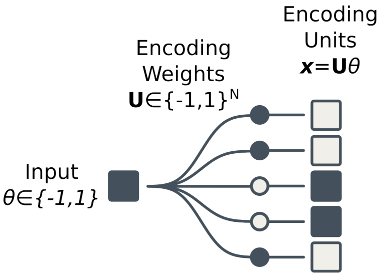


(a) Fixed Weights:
Loss of Excitability
(b) Sensitivity Homeostasis:
Punctuated Stability
(c) Hebbian Homeostasis:
Stable Readout
Firing rate (a.u.) →
Original tuning
5% reconfigured
10%
25%
50%
90%
95%
100%
0
Location
100
0
Location
100
0
Location
100

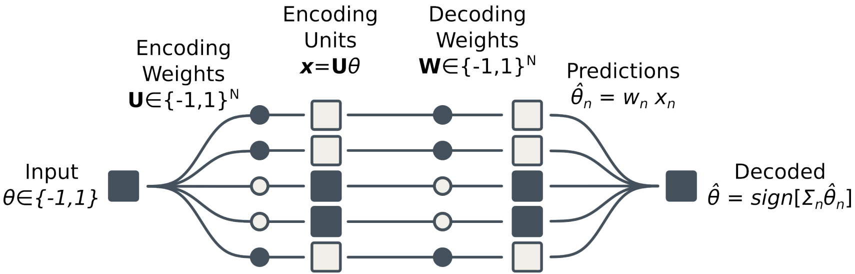
Why does this work?

Binary Threshold Analogy

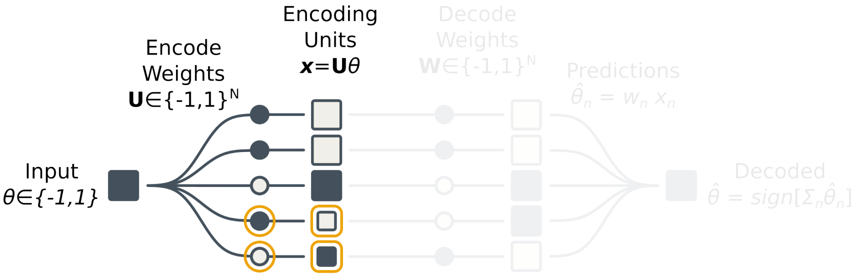
Error-Correcting Code



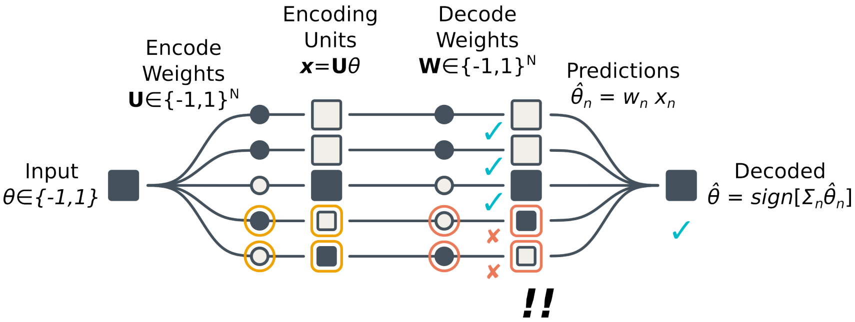
Error-Correcting Code



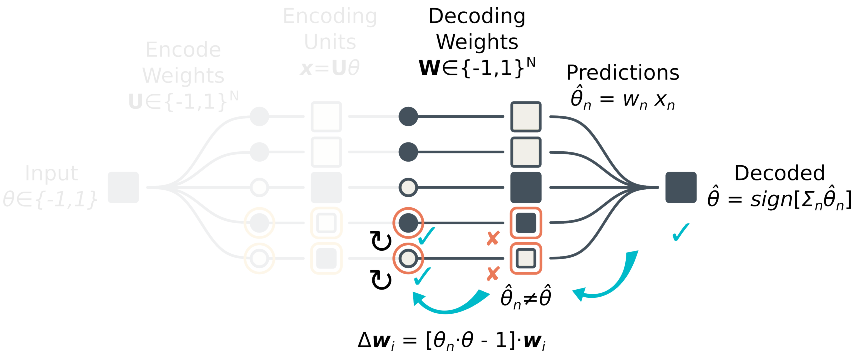
Error-Correcting Code



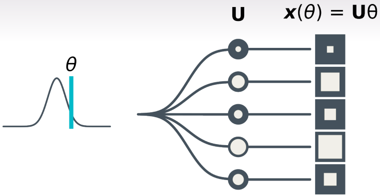
Error-Correcting Code



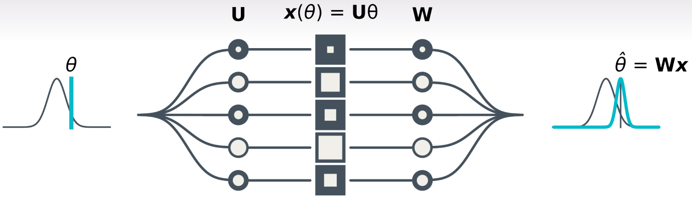
Self-Healing Code



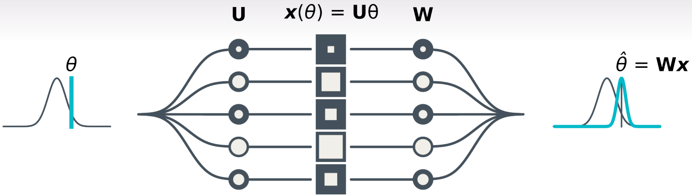
Linear Analogy

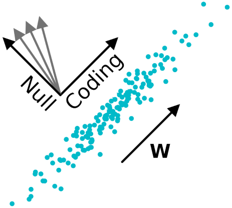


Redundant linear encoder

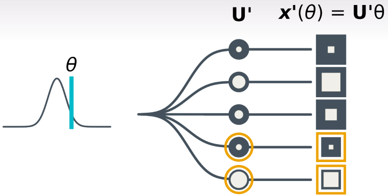


Low-D structure in High-D space

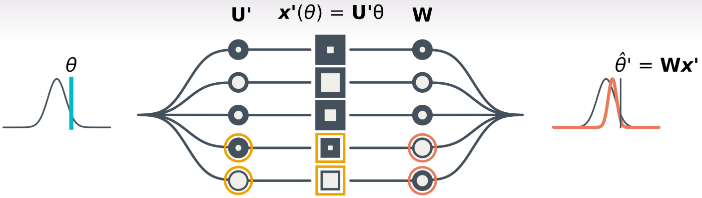




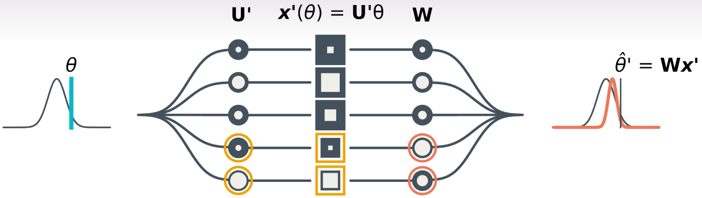
Many null-dimensions, weights align with signal dimensions

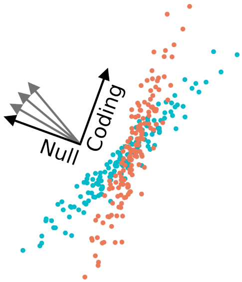


Small change in encoding...

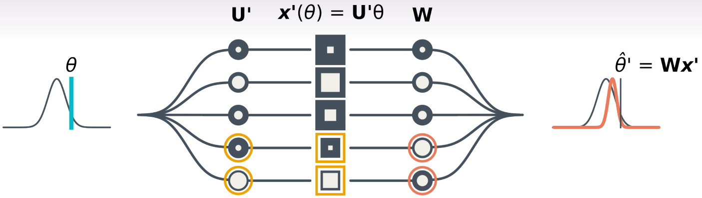


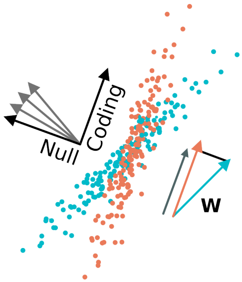
Loss of drive to readout



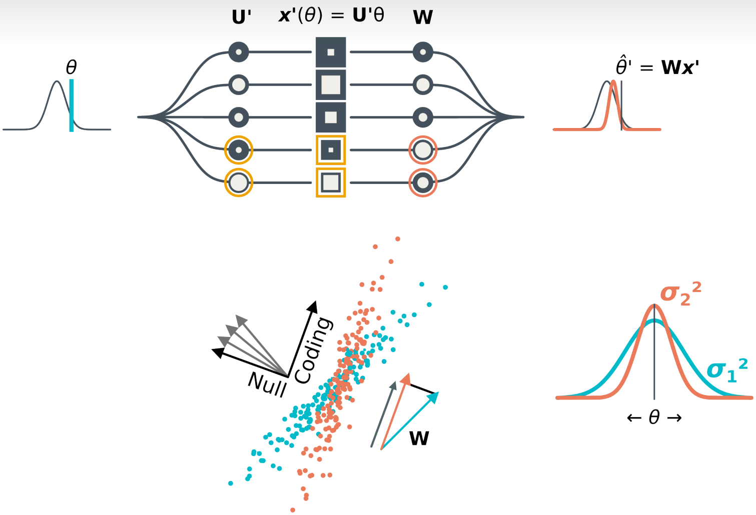


New embedding of low-D structure

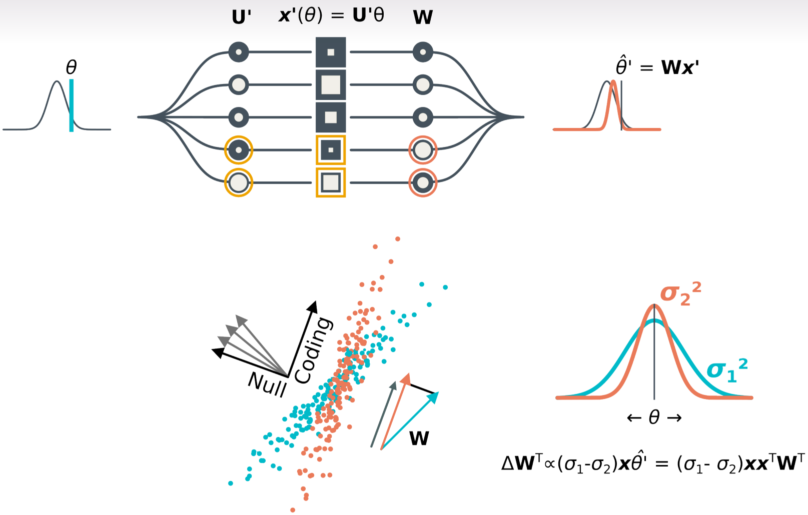




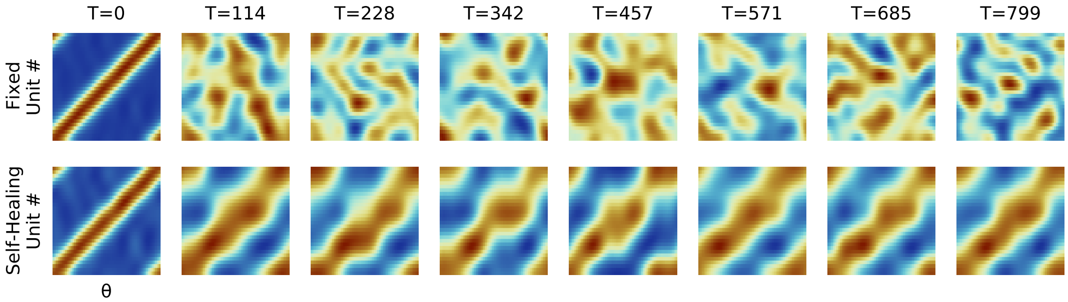
Weights no longer match signal variability

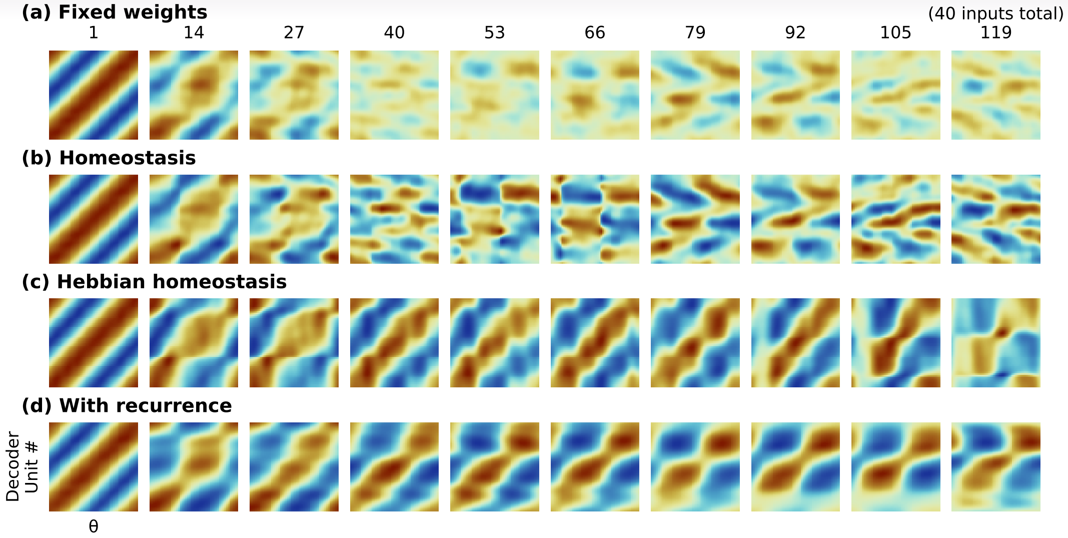


Detect loss of readout sensitivity



Hebbian homeostasis: realign weights to low-D structure


T=0
T=114
T=228
T=342
T=457
T=571
T=685
T=799
Fixed
Unit #
Self-Healing
Unit #
θ


(a) Fixed weights
(40 inputs total)
1
14
27
40
53
66
79
92
105
119
(b) Homeostasis
(c) Hebbian homeostasis
(d) With recurrence
Decoder
Unit #
θ

Linear-nonlinear readout

$$y(\theta) = \phi[\mathbf{W}\mathbf{x}(\theta)]$$

Sensitivity $y'(\theta) = \phi'[\mathbf{W}\mathbf{x}(\theta)]$
Drift $\Delta\mathbf{x}$; Average squared tuning change:

$$\left\langle \Delta_y^2 \right\rangle = \mathbf{W} \left\langle \Delta\mathbf{x}\Delta\mathbf{x}^\intercal \cdot y'(\theta)^2 \right\rangle \mathbf{W}^\intercal$$

Average sensitivity: $\|y'\|^2 = \int_{d\theta} y'(\theta)^2$
Normalized sensitivity: $\rho(\theta) = y'(\theta)^2 / \|y'\|^2$

$$\left\langle \Delta_y^2 \right\rangle = \|y'\|^2 \cdot \mathbf{W} \langle \Delta\mathbf{x}\Delta\mathbf{x}^\intercal \cdot \rho(\theta) \rangle \mathbf{W}^\intercal = \|y'\|^2 \cdot \mathbf{W} \Sigma_{\Delta x}^{\rho(\theta)} \mathbf{W}^\intercal$$

$\sim$ **Binary: saturating responses make $\|y'\|^2$ small**

$\sim$ **Linear: Locally-re-weighted input drift $\Sigma_{\Delta x}^{\rho(\theta)}$ is low rank**

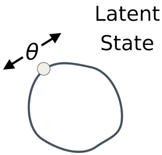
Representational drift is gradual (or null). It can be tracked via error feedback. Ongoing practice could provide this feedback, via prediction errors. However, this does not lead to stable internal representations.

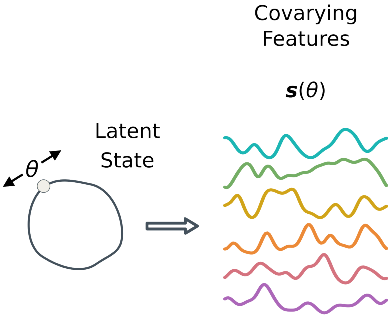
Model drift as shifting encoding weights

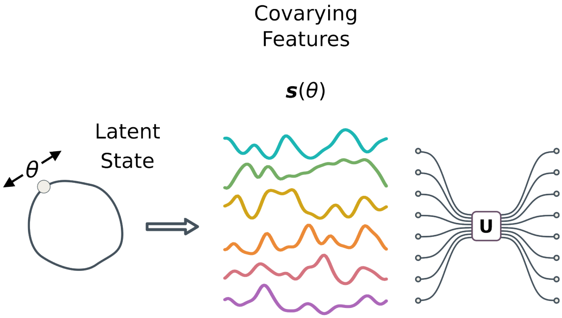
- Activity has low-D structure
- Sensitivity homeostasis leads to punctuated stability: occassional large shifts
- Hebbian homeostasis uses redundancy to re-learn weights as drift occurs
    - Binary: hard to change saturated responses
    - Linear: track low-D subspace
- Leads to stable readouts of unstable codes

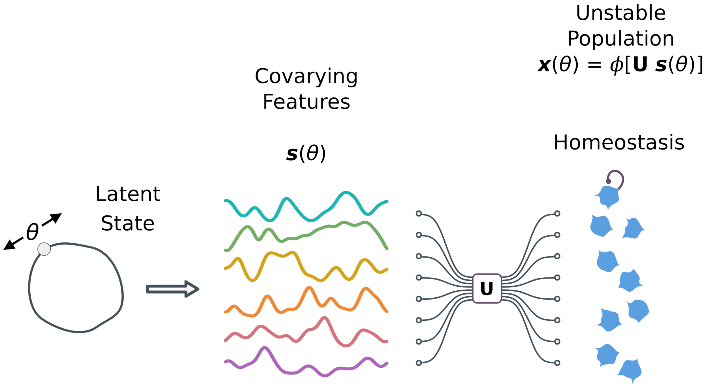
Next: Stabilizing population codes

Population Interactions

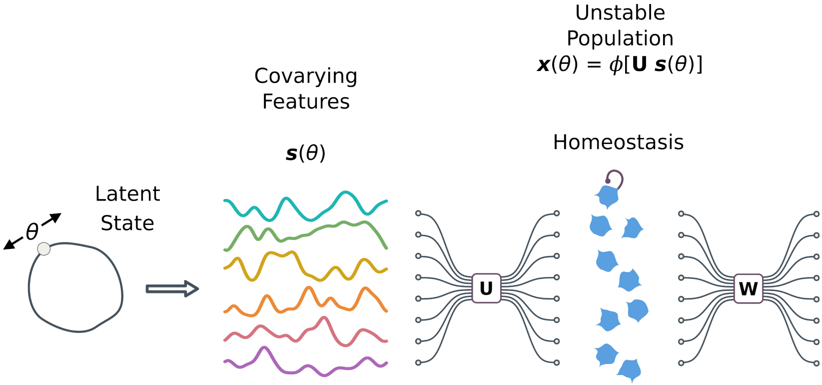

Latent
State
θ

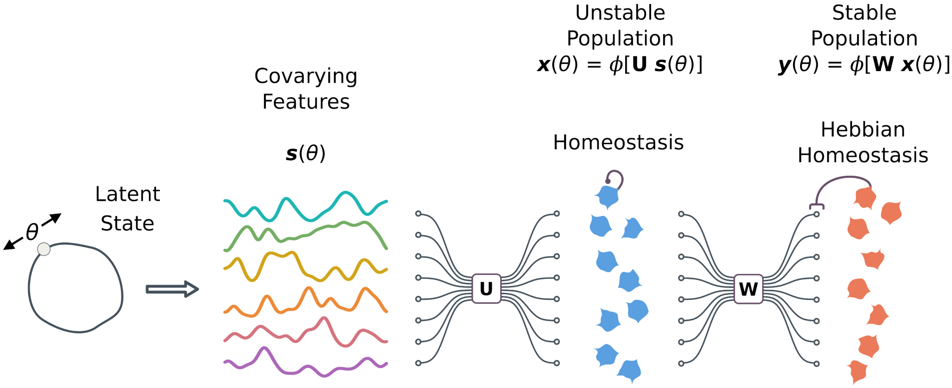

Covarying
Features
$\boldsymbol{s}(\theta)$
Latent
State
θ

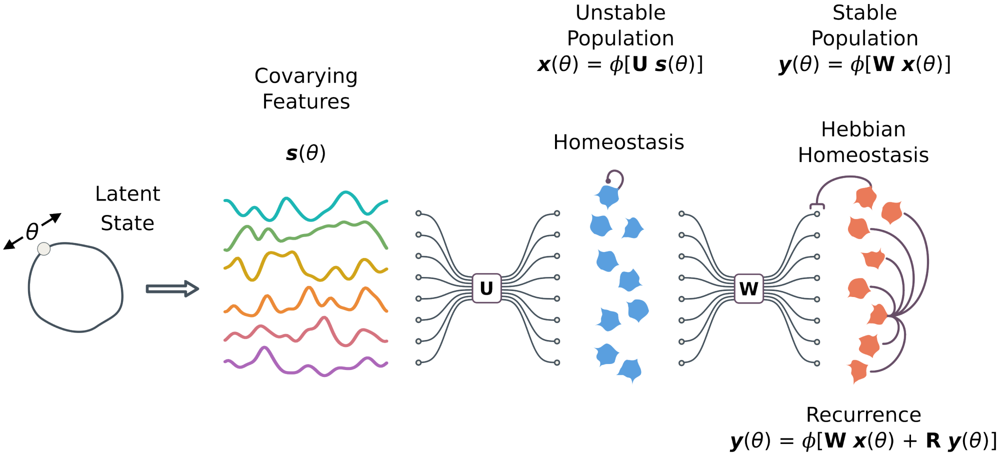

Covarying
Features
$\boldsymbol{s}(\theta)$
Latent
State
θ
U



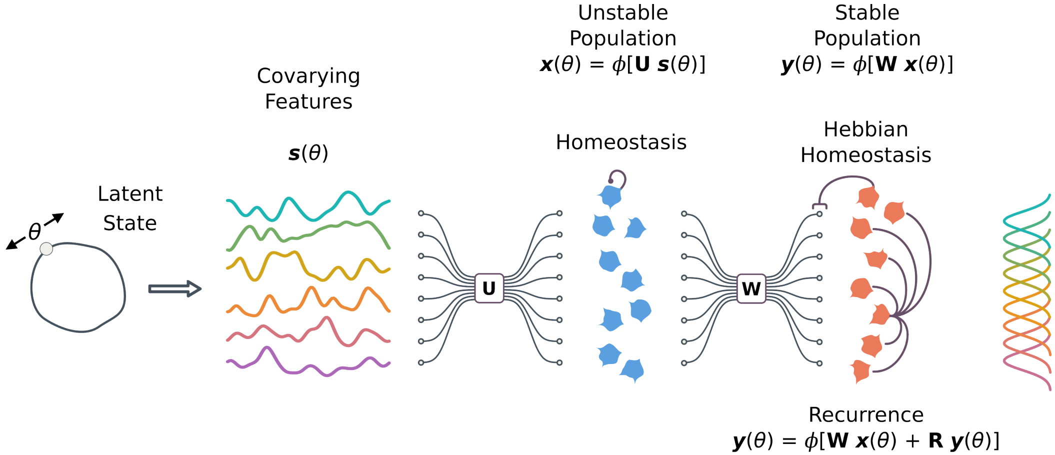
Unstable
Population
$\boldsymbol{x}(\theta) = \phi[\mathbf{U}\ \boldsymbol{s}(\theta)]$
Covarying
Features
$\boldsymbol{s}(\theta)$
Homeostasis
Latent
State
θ
U


Unstable
Population
$\boldsymbol{x}(\theta) = \phi[\mathbf{U}\,\boldsymbol{s}(\theta)]$
Covarying
Features
$\boldsymbol{s}(\theta)$
Homeostasis
Latent
State
θ
U
W

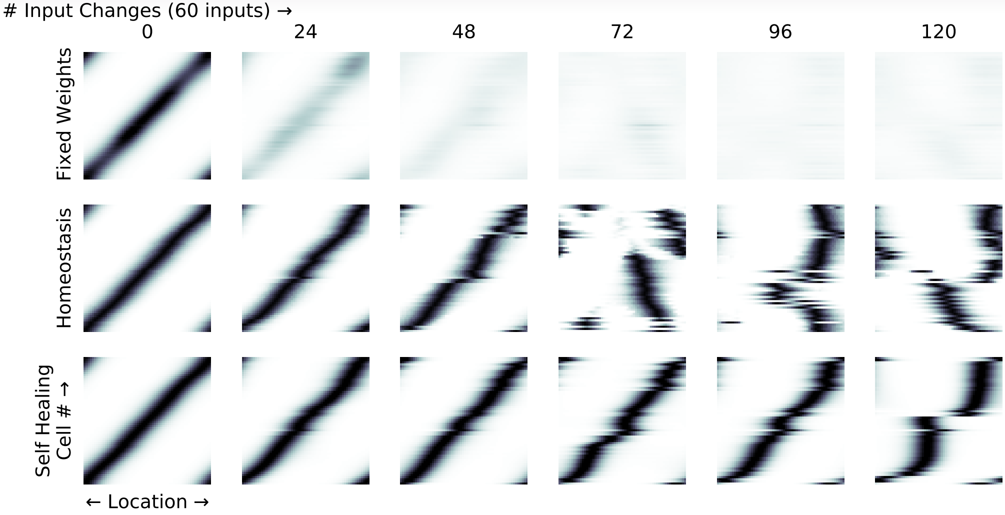

Latent
State
θ
Covarying
Features
$\boldsymbol{s}(\theta)$
Unstable
Population
$\boldsymbol{x}(\theta) = \phi[\mathbf{U}\ \boldsymbol{s}(\theta)]$
Homeostasis
U
Stable
Population
$\boldsymbol{y}(\theta) = \phi[\mathbf{W}\ \boldsymbol{x}(\theta)]$
Hebbian
Homeostasis
W



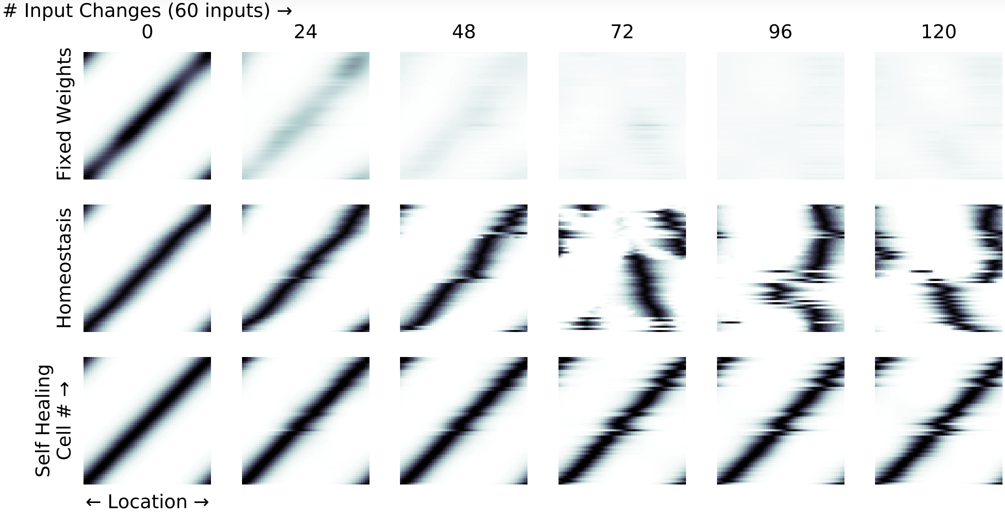
Latent
State
θ
Covarying
Features
$\boldsymbol{s}(\theta)$
Unstable
Population
$\boldsymbol{x}(\theta) = \phi[\mathbf{U}\ \boldsymbol{s}(\theta)]$
Homeostasis
U
Stable
Population
$\boldsymbol{y}(\theta) = \phi[\mathbf{W}\ \boldsymbol{x}(\theta)]$
Hebbian
Homeostasis
W
Recurrence
$\boldsymbol{y}(\theta) = \phi[\mathbf{W}\ \boldsymbol{x}(\theta) + \mathbf{R}\ \boldsymbol{y}(\theta)]$


Unstable
Population
$\boldsymbol{x}(\theta) = \phi[\mathbf{U}\ \boldsymbol{s}(\theta)]$
Stable
Population
$\boldsymbol{y}(\theta) = \phi[\mathbf{W}\ \boldsymbol{x}(\theta)]$
Covarying
Features
$\boldsymbol{s}(\theta)$
Homeostasis
Hebbian
Homeostasis
Latent
State
θ
U
W
Recurrence
$\boldsymbol{y}(\theta) = \phi[\mathbf{W}\ \boldsymbol{x}(\theta) + \mathbf{R}\ \boldsymbol{y}(\theta)]$

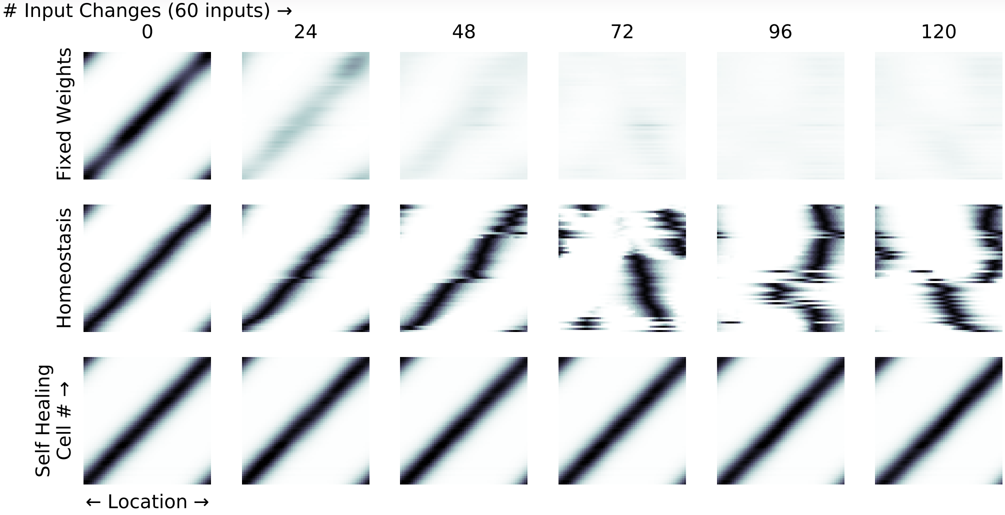
Population without Competition



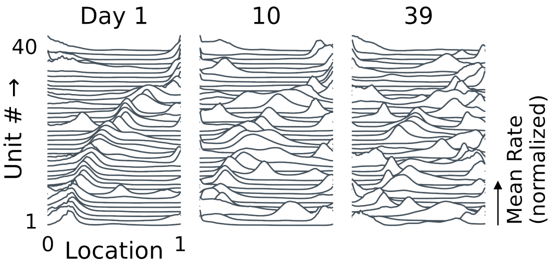
Population with Competition



Population with Competition and Recurrence

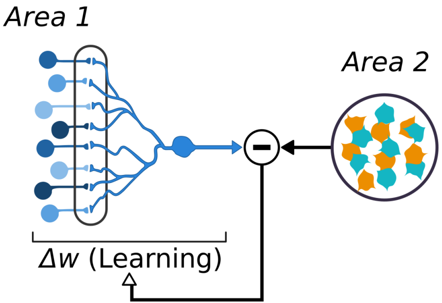


Drift is gradual/null:



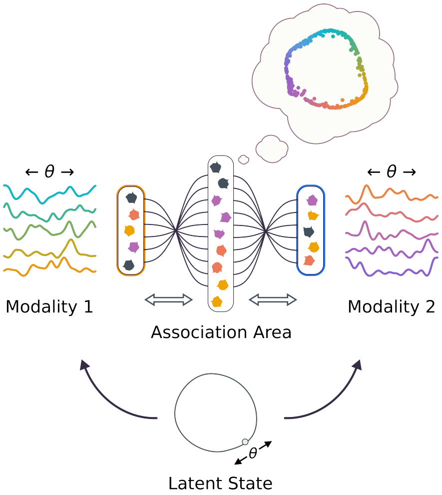
Drift is gradual/null:

- Track with error feedback



Drift is gradual/null:

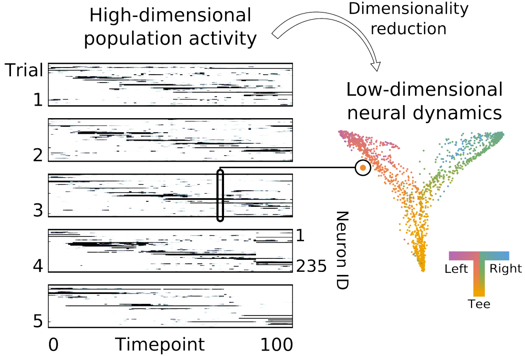
- Track with error feedback
- Ongoing practice provides this



Drift is gradual/null:

- Track with error feedback
- Ongoing practice provides this

Model drift: Inputs have low-D structure

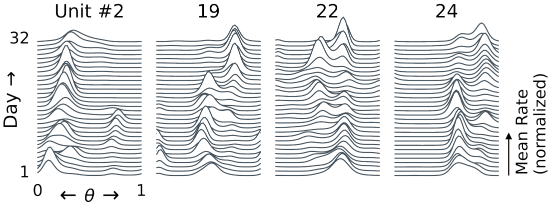


Drift is gradual/null:

- Track with error feedback
- Ongoing practice provides this

Model drift: Inputs have low-D structure

- Hard to change tuning; punctuated stability



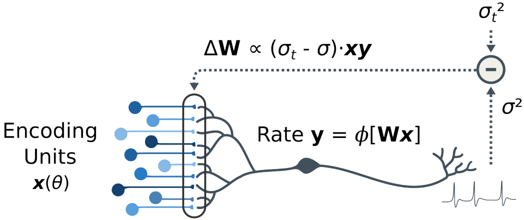
Drift is gradual/null:

- Track with error feedback
- Ongoing practice provides this

Model drift: Inputs have low-D structure

- Hard to change tuning; punctuated stability

Hebbian homeostasis:



Drift is gradual/null:

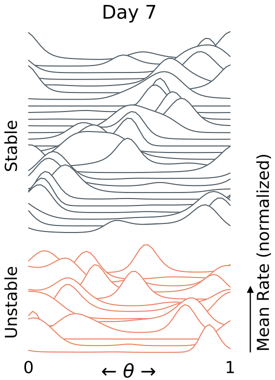
- Track with error feedback
- Ongoing practice provides this

Model drift: Inputs have low-D structure

- Hard to change tuning; punctuated stability

Hebbian homeostasis:

- Re-learn tuning as inputs change



Drift is gradual/null:

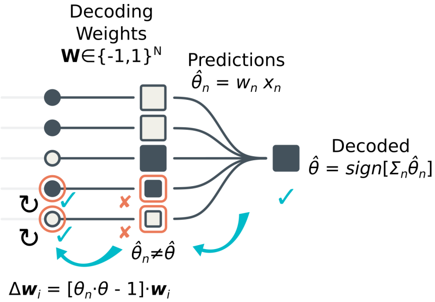
- Track with error feedback
- Ongoing practice provides this

Model drift: Inputs have low-D structure

- Hard to change tuning; punctuated stability

Hebbian homeostasis:

- Re-learn tuning as inputs change
- Binary: hard to change saturated responses



(remember)

Drift is gradual/null:

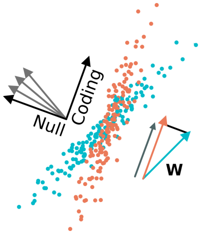
- Track with error feedback
- Ongoing practice provides this

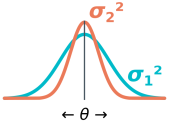
Model drift: Inputs have low-D structure

- Hard to change tuning; punctuated stability

Hebbian homeostasis:

- Re-learn tuning as inputs change
- Binary: hard to change saturated responses
- Linear: track low-D subspace





$$\Delta \mathbf{W}^{\top} \propto (\sigma_1 - \sigma_2)\boldsymbol{x}\hat{\theta}' = (\sigma_1 - \sigma_2)\boldsymbol{x}\boldsymbol{x}^{\top}\mathbf{W}^{\top}$$

Drift is gradual/null:

- Track with error feedback
- Ongoing practice provides this

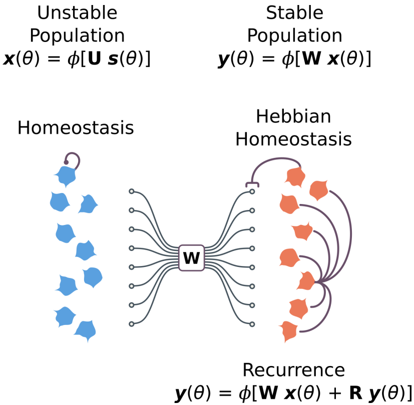
Model drift: Inputs have low-D structure

- Hard to change tuning; punctuated stability

Hebbian homeostasis:

- Re-learn tuning as inputs change
- Binary: hard to change saturated responses
- Linear: track low-D subspace

Population interactions



Drift is gradual/null:

- Track with error feedback
- Ongoing practice provides this

Model drift: Inputs have low-D structure

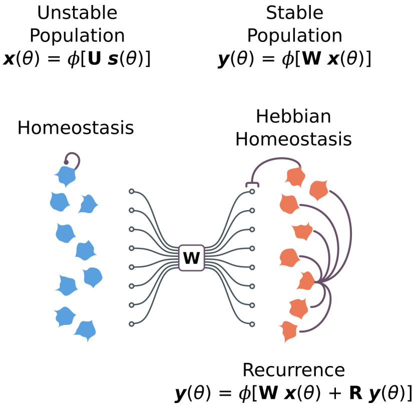
- Hard to change tuning; punctuated stability

Hebbian homeostasis:

- Re-learn tuning as inputs change
- Binary: hard to change saturated responses
- Linear: track low-D subspace

Population interactions

- Normalize: competition ensures coverage



(remember)

Drift is gradual/null:

- Track with error feedback
- Ongoing practice provides this

Model drift: Inputs have low-D structure

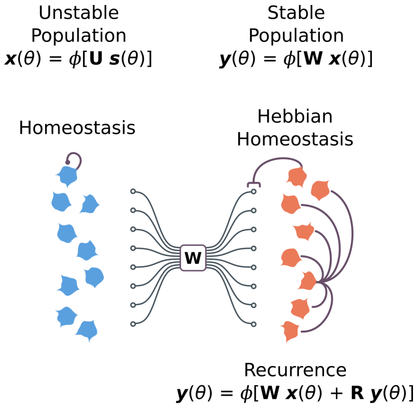
- Hard to change tuning; punctuated stability

Hebbian homeostasis:

- Re-learn tuning as inputs change
- Binary: hard to change saturated responses
- Linear: track low-D subspace

Population interactions

- Normalize: competition ensures coverage
- Recurrent connections → stable readout



End of Content

Thanks to:

Dhruva Raman

Timothy O'Leary

Chris Harvey

Laura Driscoll

Adrianna Loback

Fulvio Forni

Alon Rubin

Yaniv Ziv

Aspects of this work published in:

Rule ME, Loback AR, Raman DV, Driscoll L, Harvey CD, O'Leary T. 2020. Stable task information from an unstable neural population. eLife

Rule ME, O'Leary T, Harvey CD. 2019. Causes and consequences of representational drift. Current opinion in neurobiology 58:141–147

Funding:

This work was supported by the Human Frontier Science Program (RGY0069), ERC Starting Grant (StG FLEXNEURO 716643) and grants from the NIH (NS089521, MH107620, NS108410)