TO UNTANGLE MULTI-CIRCUIT NEURAL DYNAMICS, ALL YOU NEED IS LAG

Anonymous authors
Paper under double-blind review

ABSTRACT

The brain computes through interactions between neural populations that shape how their activity evolves over time. These interactions often span multiple brain regions, thus forming brain-wide networks that reconfigure as the animal shifts task contexts, behaviors, and decisions. Within these networks, neural signals travel at finite speed, so one population drives another only after a delay whose length reflects both the wiring involved and the animal's current state. Dynamical systems are a natural way to model such lagged interactions, yet existing models often assume a fixed lag, allow several lags but fix the interaction structure in time, or permit richer lag structure at the cost of uninterpretable dynamics. Here, we present Multi-Lag decomposed Linear Dynamical Systems (ML-dLDS), a framework that captures how time-varying neural dynamics arise from multiple co-occurring processes acting across several lags. ML-dLDS represents neural activity as a sparse decomposition of core neural interactions, or *circuits*, which are reused over time and act across a sparse set of lags. Each circuit-lag pair carries its own time-varying coefficient that sets its momentary contribution to the overall network dynamics, which thereby shift from moment to moment as the active circuit combination changes. We fit the circuits, their per-lag gains, and their time-varying coefficients through an alternating optimization under structured-sparsity penalties. We first validate ML-dLDS on synthetic datasets, where it recovers the ground-truth circuits and their lags, and then demonstrate it on multi-region electrophysiological recordings, where it reveals lag structure that single-lag models oversimplify. Finally, we apply ML-dLDS to COVID-19 case data, where it uncovers interpretable multi-lag structure across countries.

1 INTRODUCTION

A central goal of systems neuroscience is to understand how populations of neurons interact to produce behavior. These interactions are dynamic, reshaped over time by mechanisms including synaptic plasticity and neuromodulation (Abbott & Nelson, 2000; Marder, 2012) as the brain moves through different behaviors, decisions, and task conditions. Populations across the brain drive one another faster or slower depending on the pathway and the areas involved. For example, feedforward signaling from V1 to V2 acts within a few milliseconds, while feedback between the same areas travels through different population activity patterns at a slower lag, and this balance shifts over the trial (Semedo et al., 2022); in primate cortex, interaction delays grow in a task-dependent way, from near-zero within somatosensory areas to roughly 80-140 ms for premotor and motor areas (Tauste Campo et al., 2015); across the human cortex, inter-regional lags themselves fluctuate over time rather than holding to one fixed value (Moon et al., 2022). These examples suggest that neural interactions operate across a range of lags that shift with pathway, task conditions, and time.

Dynamical-systems models describe how a system evolves over time by relating its current state to past states at a given lag, and thereby offer a central tool for describing neural population activity and its underlying interactions. An ideal dynamical model of neural activity should therefore both capture the time-varying property of neural dynamics and allow interactions to operate across multiple lags. In addition, for a model to be scientifically useful, its components should admit interpretation in terms of the specific neural interactions they represent.

Linear dynamical systems (LDSs) offer a natural route to this interpretability, since each component reads as an interaction, and time-varying LDS variants such as switching or decomposed mod-

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

els (Linderman et al., 2016; Mudrik et al., 2024a) carry it to changing dynamics. Yet these models rely on a single, predefined lag, which forces a fast local interaction and a slow one into a shared term. More expressive deep-learning methods like LFADS (Pandarinath et al., 2018) carry information across many past time steps through a recurrent state, but they do not represent explicit, separable lags, and the dynamics they produce are typically uninterpretable.

Here, we introduce ML-dLDS (Multi-Lag decomposed Linear Dynamical Systems), which learns interpretable dynamics that unfold across multiple lags at once. The model shares a single set of circuits across all lags, so it stays compact even as the number of lags grows. A per-lag gain sets how strongly each circuit acts at each lag, yielding a sparse lag profile that represents the few active lags a circuit can use. These profiles are assumed distinct across circuits, so different circuits occupy different lags and thereby capture interactions at multiple timescales. Time-varying coefficients then activate the circuits, so the overall dynamics change over time as the combination of active circuits shifts, with the lags at which each circuit acts revealing the interaction timescale.

Our contributions include:

- We introduce ML-dLDS, which models time-varying neural dynamics as a sparse set of core interactions shared across multiple co-occurring lags, keeping the representation compact as lags grow.
- ML-dLDS learns these interactions in an interpretable form, recovering the timescale of each and thereby separating fast from slow communication.
- We show that ML-dLDS recovers ground-truth interactions, lags, and activations on synthetic data, reveals lag structure single-lag models miss on multi-region recordings, and captures interpretable multi-lag structure on COVID-19 case data.

2 Related Work

Prior methods for neural population dynamics either act at a single lag, multiply parameters across lags, or trade interpretability for expressive power. Deep-learning models capture neural population dynamics with few structural assumptions (Pandarinath et al., 2018; Ye & Pandarinath, 2021). Data-constrained recurrent networks push this further and read directed interactions between regions off the fitted weights (Perich et al., 2021). These models typically fit population activity well, but their components do not map onto interpretable interactions.

Interpretable models instead describe the dynamics through a few simple components that link back to circuit mechanisms. Switching models piece the dynamics together from discrete states of locally simple regimes, spanning linear (Linderman et al., 2016), multi-region (Glaser et al., 2020), and nonlinear (Karniol-Tambour et al., 2024) variants. While switching captures time-varying dynamics in an interpretable way, its one-regime-at-a-time structure misses interactions that unfold together. Decomposition models instead express the dynamics as a sparse, time-varying combination of shared circuits that can act at once (Mudrik et al., 2024a; Chen et al., 2024), with a multi-region extension that ties each circuit to the populations it couples (Mudrik et al., 2024b). Both, however, assume a single fixed lag, so a mechanism acting quickly and one acting slowly become indistinguishable within the model.

A separate line of work allows interactions at more than one lag. Vector autoregression predicts the current state as a sum of separate learned matrices, one per past lag (Stock & Watson, 2001), and time-varying variants let those matrices drift over the recording (Harris et al., 2021), but both keep a separate circuit per lag and share none across them. Generalized linear models act across many lags through spike-history and coupling filters (Pillow et al., 2008), but fix these per unit and never infer the lag itself. Switching low-rank tensor models factor a per-lag autoregressive tensor (Lee et al., 2023), but allow one active state at a time. The lagged form also echoes delay-differential equations and delay-line networks (Zhu et al., 2021; Waibel et al., 1989), which carry past states forward without interpretable interactions. None of these models reuses a circuit across lags or recovers which lag each interaction acts at.

A different approach assigns a delay to latent signals shared across regions and recovers that delay from data, using Gaussian-process latents whose cross-region covariance is shifted in time (Gokcen et al., 2022; 2024; Li et al., 2024a), with recent work letting that delay vary over the recording (Li et al., 2024b). These models are useful for showing how regions align in time, yet they recover delayed latent signals rather than the dynamical operator that generates them.

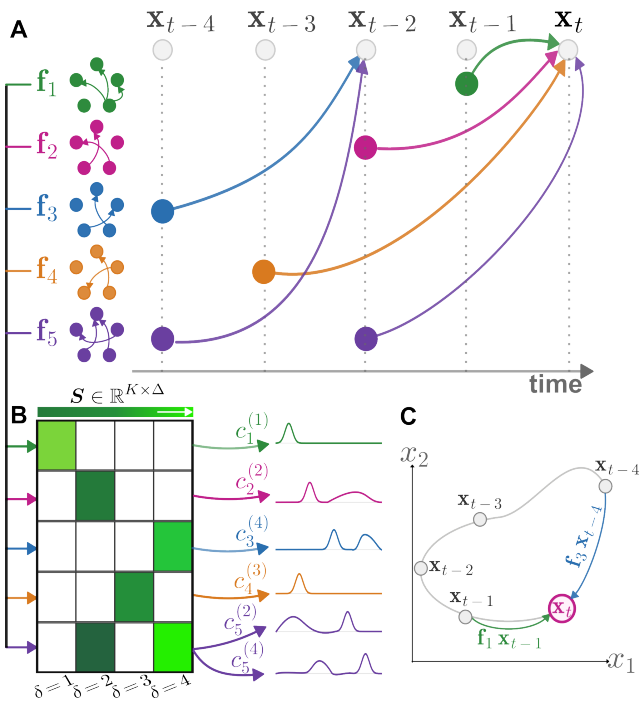

Figure 1: **Schematic. ML-dLDS decomposes the dynamics into shared circuits, a sparse lag map, and per-lag activations,** following $\boldsymbol{x}_t = \sum_\delta \sum_k c_{kt}^{(\delta)}\, s_k^{(\delta)}\, \boldsymbol{f}_k\, \boldsymbol{x}_{t-\delta}$. **(A)** The model builds each state $\boldsymbol{x}_t$ from past states $\boldsymbol{x}_{t-\delta}$ through K time-fixed circuits $\{\boldsymbol{f}_k\}_{k=1}^K$ (left). Each $\boldsymbol{f}_k \in \mathbb{R}^{P\times P}$ shapes $\boldsymbol{x}_t$ from past states at specific lags $\delta \in \{1,\dots,\Delta\}$. A circuit can act at several lags, shown as arrows of the same color (e.g., for $\boldsymbol{f}_5$). **(B)** The sparse lag map $\boldsymbol{S} \in \mathbb{R}^{K\times\Delta}$ sets the lags each circuit can use, with entry $s_k^{(\delta)}$ its gain at lag δ (colorbar weak to strong). Each nonzero cell (k,δ) carries its own time-varying activation $c_{kt}^{(\delta)}$ (right), e.g., $\boldsymbol{f}_5$ is associated with two traces, $c_{5t}^{(2)}$ and $c_{5t}^{(4)}$. **(C)** View of the dynamics in the x_1 vs x_2 space, where each circuit maps a past point onto $\boldsymbol{x}_t$ and δ sets how far back it reaches.

3 Our Method, ML-dLDS

3.1 Problem Formulation

Consider the activity of N neural units recorded over T time steps as a time series $\boldsymbol{Y} \in \mathbb{R}^{N\times T}$, where each column $\boldsymbol{y}_{:,t} \in \mathbb{R}^N$ holds the population activity at time $t = 1,\dots,T$. We assume that a lower-dimensional, unobserved latent state $\boldsymbol{x}_t$ drives the observed neural activity $\boldsymbol{y}_t$ via some mapping $\boldsymbol{x}_t \mapsto \boldsymbol{y}_t$. This latent state evolves over time in a possibly complex, time-varying manner through an unknown dynamical rule $\boldsymbol{x}_t = \boldsymbol{F}_t(\{\boldsymbol{x}_{t-\delta}\}_{\delta=1}^{\Delta})$ that ties each state to those before it. The dynamical rule $\boldsymbol{F}_t$ may change over time to reflect evolving behavior and internal state, and may combine influences from several past time lags. Here, our goal is to recover, from $\boldsymbol{Y}$ alone, the recurring interactions that drive the dynamics, the lags they act at, and how strongly each is engaged over time.

3.2 Model

Our model consists of two parts, (1) how the observed activity arises from the latent state, and (2) how the latent state evolves over time.

Observation model. We assume each recorded neuron belongs to one or more of P neural populations, with membership weights given by row n of a loading matrix $\boldsymbol{A} \in \mathbb{R}^{N\times P}$, such that $\boldsymbol{y}_t = \boldsymbol{A}\boldsymbol{x}_t + \boldsymbol{\epsilon}_t$, where $\boldsymbol{\epsilon}_t \sim \mathcal{N}(\boldsymbol{0}, \sigma^2\boldsymbol{I})$ is the observation noise. Notably, we keep this map linear so that each latent direction stays an interpretable population pattern.

Circuits, lags, and activations. To capture the dynamics, we assume the evolution of the latent state emerges from K neural circuits $\{\boldsymbol{f}_k\}_{k=1}^K$ that can act simultaneously or independently. Each circuit $\boldsymbol{f}_k \in \mathbb{R}^{P\times P}$ is a fixed interaction pattern between latents, such that $\boldsymbol{f}_k[i,j]$ represents the influence of latent j on latent i. Each circuit acts with a delay, reaching back some number of steps δ, sometimes several at once, to shape the present state $\boldsymbol{x}_t$ from past ones $\boldsymbol{x}_{t-\delta}$. We define a matrix $\boldsymbol{S} \in \mathbb{R}^{K\times\Delta}$ that collects these reaches, with its entry $s_k^{(\delta)} \geq 0$ setting how strongly circuit k acts at lag δ, so that $s_k^{(\delta)}$ reads as the timescale at which the circuit acts. We further assume each circuit contributes to the dynamics in a time-varying way, through a coefficient $c_{kt}^{(\delta)}$ that sets how strongly circuit k drives the target state $\boldsymbol{x}_t$ from the source state $\boldsymbol{x}_{t-\delta}$ (Fig. 1C).

Dynamics model. Together, the contribution of a single circuit $\boldsymbol{f}_k$ at one lag δ to the target state $\boldsymbol{x}_t$ is

$$\boldsymbol{g}_{kt}^{(\delta)} = \underbrace{c_{kt}^{(\delta)}\, s_k^{(\delta)}\, \boldsymbol{f}_k}_{\text{circuit } k \text{ weighted at lag } \delta} \boldsymbol{x}_{t-\delta}. \tag{1}$$

The above product keeps the circuit $\boldsymbol{f}_k$ and its weight $c_{kt}^{(\delta)} s_k^{(\delta)}$ separate, so the same interaction can act at different strengths. Summing over all circuits and lags gives the state update

$$\boldsymbol{x}_t = \sum_{\delta=1}^{\Delta} \sum_{k=1}^{K} \boldsymbol{g}_{kt}^{(\delta)} = \sum_{\delta=1}^{\Delta} \sum_{k=1}^{K} \underbrace{c_{kt}^{(\delta)}}_{\text{activation}} \underbrace{s_k^{(\delta)}}_{\text{lag gain}} \underbrace{\boldsymbol{f}_k}_{\text{interaction}} \underbrace{\boldsymbol{x}_{t-\delta}}_{\text{source state}}. \tag{2}$$

Parsimony via structured sparsity. We further assume the dynamics are structured, so that each circuit acts at only a few lags and only a few circuits are active at any moment. These assumptions serve two goals, (1) they keep the dynamics description interpretable, with each state drawing on a small set of circuit-lag pairs rather than a dense mixture, and (2) they resolve the ambiguity of the fit, where many activation patterns would otherwise reproduce the same $\boldsymbol{x}_t$ (Sec. 3.3).

Formally, these assumptions place structured sparsity on both the lag gains $\boldsymbol{S}$, which applies to each circuit's lag profile $\boldsymbol{S}_{k,:}$ so that each circuit acts at only a few lags, and the coefficients $\boldsymbol{c}$, which applies to the activation profile $\boldsymbol{c}_t$ so that only a few circuits are active at each time point and each state draws on a small set of circuits. We emphasize that the two assumptions are complementary: $\boldsymbol{S}$ is time-invariant, while $\boldsymbol{c}$ varies over time, and this split is what keeps each circuit interpretable as acting at a fixed sparse set of lags.

One case worth emphasizing is when two past lags carry nearly identical values, which can make it ambiguous which lag is the true source. ML-dLDS handles this by sharing $\boldsymbol{S}$ across time points, so even when two states (e.g. $\boldsymbol{x}_{t-2}$ and $\boldsymbol{x}_{t-4}$) are equally plausible sources at one moment, $\boldsymbol{S}$ captures the correct lag structure from the whole recording and settles on the delay that actually drives the overall dynamics. A harder case arises when past states stay ambiguous across the whole recording, which happens when two lags remain equally good candidates at every time point (e.g. in an oscillatory regime). Here $\boldsymbol{S}$-sharing cannot break the tie. ML-dLDS handles this by favoring the shortest lag as the most parsimonious choice when this ambiguity arises (App. J).

3.3 Fitting

Fitting ML-dLDS means learning several coupled unknowns, including the latents, circuits, as well as their time-varying coefficients and lag profiles. Since each circuit predicts from several past states, they couple across lags in complex ways, calling for a new fitting procedure that recovers the components jointly by alternating minimization under the structured sparsity considerations described above. Below, we first specify the two sparsity penalties, on the coefficients $\boldsymbol{c}$ and on the lag gains $\boldsymbol{S}$, then combine them with the data terms into a single objective, and finally describe the alternating updates that minimize it.

Coefficient sparsity. The coefficient sparsity is enforced by an ℓ_1 penalty on $\boldsymbol{c}$ during inference,

$$\mathcal{R}_c = \lambda_c \sum_t \sum_\delta \|\boldsymbol{c}_t^{(\delta)}\|_1, \tag{3}$$

where λ_c sets the sparsity strength and the sum over t and δ counts every active circuit-lag pair.

Lag sparsity. The lag sparsity is enforced by a lag-weighted ℓ_1 penalty on $\boldsymbol{S}$ that drives each circuit onto few lags, plus an orthogonality term that pushes circuits onto different lags,

$$\mathcal{R}_S = \lambda_1 \underbrace{\sum_k \sum_\delta w_\delta\, s_k^{(\delta)}}_{\text{lag sparsity}} + \lambda_2 \underbrace{\sum_{k \neq k'} \langle \boldsymbol{S}_{k,:}, \boldsymbol{S}_{k',:} \rangle^2}_{\text{orthogonality}} + \lambda_3 \underbrace{\|\boldsymbol{S}\|_F^2}_{\text{stability}}, \tag{4}$$

where w_δ grows with δ, so longer lags cost more and a circuit settles on its shortest lag when several fit equally, and uniform weights $w_\delta = 1$ are a special case that reduces to plain ℓ_1.

The orthogonality term $\sum_{k \neq k'} \langle \boldsymbol{S}_{k,:}, \boldsymbol{S}_{k',:} \rangle^2$ spreads circuits across different lags rather than piling them on the same one, and the last ℓ_2 term $\|\boldsymbol{S}\|_F^2$ ensures a stable convex fit. The weights λ_1, λ_2, and λ_3 are hyperparameters that set the strength of the lag-sparsity, orthogonality, and ℓ_2 terms.

Objective. The overall objective combines the two fidelity terms with the sparsity penalties,

$$\mathcal{L} = \sum_t \underbrace{\|\boldsymbol{y}_t - \boldsymbol{A}\boldsymbol{x}_t\|_2^2}_{\text{observation fidelity}} + \lambda_{\text{dyn}} \sum_t \underbrace{\Big\|\boldsymbol{x}_t - \sum_{\delta=1}^{\Delta}\sum_{k=1}^{K} c_{kt}^{(\delta)}\, s_k^{(\delta)}\, \boldsymbol{f}_k\, \boldsymbol{x}_{t-\delta}\Big\|_2^2}_{\text{dynamics fidelity}} + \mathcal{R}_c + \mathcal{R}_S, \tag{5}$$

where the first term ties latents to observations, the second to prediction, and λ_{dyn} balances them.

Optimization. We initialize the circuits from a warm start built on detected simple segments (App. A), then minimize equation 5 by alternating minimization, updating each of $\boldsymbol{x}$, $\boldsymbol{c}$, $\boldsymbol{S}$, and $\{\boldsymbol{f}_k\}$ in turn with the others fixed (Alg. 1).

Infer latents and activations. Given $\boldsymbol{S}$ and $\{\boldsymbol{f}_k\}$, we co-infer the latent state and activations at each time point, solving for $\boldsymbol{x}_t$ and $\boldsymbol{c}_t$ together with an ℓ_1 penalty on $\boldsymbol{c}_t$ via SPGL1, an ℓ_1-regularized least-squares solver (Van Den Berg & Friedlander, 2009; van den Berg & Friedlander, 2019),

$$\boldsymbol{x}_t, \boldsymbol{c}_t \leftarrow \arg\min_{\boldsymbol{x}_t,\, \boldsymbol{c}_t \geq 0} \|\boldsymbol{y}_t - \boldsymbol{A}\boldsymbol{x}_t\|_2^2 + \lambda_{\text{dyn}} \Big\|\boldsymbol{x}_t - \sum_{\delta=1}^{\Delta}\sum_{k=1}^{K} c_{kt}^{(\delta)}\, s_k^{(\delta)}\, \boldsymbol{f}_k\, \boldsymbol{x}_{t-\delta}\Big\|_2^2 + \lambda_c \|\boldsymbol{c}_t\|_1. \tag{6}$$

Update lag scaling. With the other unknowns held fixed, $\boldsymbol{S}$ enters the prediction linearly, so this step forms an elastic net that we recast as a LASSO (Tibshirani, 1996) and solve with SPGL1,

$$\boldsymbol{S} \leftarrow \arg\min_{\boldsymbol{S} \geq 0}\ \lambda_{\text{dyn}} \sum_t \Big\|\boldsymbol{x}_t - \sum_{\delta=1}^{\Delta}\sum_{k=1}^{K} c_{kt}^{(\delta)}\, s_k^{(\delta)}\, \boldsymbol{f}_k\, \boldsymbol{x}_{t-\delta}\Big\|_2^2 + \mathcal{R}_S. \tag{7}$$

Update circuits. Finally, we fit each circuit $\boldsymbol{f}_k$ by ridge-regularized least squares over all its active lags, updating the circuits one at a time in Gauss-Seidel order (Golub & Van Loan, 2013),

$$\boldsymbol{f}_k \leftarrow \arg\min_{\boldsymbol{f}_k} \sum_t \Big\|\boldsymbol{r}_t^{(k)} - \sum_{\delta \in \mathcal{D}_k} c_{kt}^{(\delta)}\, s_k^{(\delta)}\, \boldsymbol{f}_k\, \boldsymbol{x}_{t-\delta}\Big\|_2^2 + \mu \|\boldsymbol{f}_k\|_F^2, \tag{8}$$

where $\mathcal{D}_k = \{\delta : s_k^{(\delta)} > 0\}$ are the lags circuit k can operate on, $\boldsymbol{r}_t^{(k)}$ is the residual with the other circuits removed, and μ is a small ridge weight that keeps the solve well-posed.

The loading $\boldsymbol{A}$ is re-estimated each outer pass by ridge least-squares with a polar projection to orthonormal columns, and these updates are cycled to convergence, with the schedule in App. B. Alg. 1 and Fig. 1 summarize and illustrate the method, with runtime and scalability in App. K and Tab. 4.

4 Experiments

We validate ML-dLDS on two synthetic datasets, multi-region neural recordings, and COVID data.

4.1 ML-dLDS recovers circuits, delays, and coefficients

For the first synthetic dataset we generated $P{=}6$ dimensions, $K{=}3$ orthogonal circuits, $\Delta{=}3$ delays, and $T{=}900$ time points. The coefficients $\boldsymbol{c}$ stay positive and fluctuate fast under a schedule that activates circuits alone and then in mixtures (additional data details are in App. E). When we apply ML-dLDS to the data, it recovers the circuits ($\text{corr}(\boldsymbol{f}, \hat{\boldsymbol{f}}) = 1.0$) and their lags, reaching a relative ℓ_2 reconstruction error of 2×10^{-4} (Fig. 2A-D). This recovery holds across seeds, with $\text{corr}(\boldsymbol{f}, \hat{\boldsymbol{f}}) = 0.96 \pm 0.08$ and lag support $97\% \pm 10\%$ over 100 seeds (App. D.2).

We then compare ML-dLDS against baselines, where it outperforms single-lag dLDS, full VAR, rSLDS, and SLDS on both reconstruction and circuit correlation (Fig. 2E, baselines information in App. C). The single-lag and VAR baselines reach low reconstruction error but low circuit correlation, which suggests they recover the coarse dynamics but miss the circuit structure. The switching models rSLDS and SLDS reach higher reconstruction error and lower circuit correlation, which we attribute to their one-mode-at-a-time form that cannot represent circuits acting together.

To test ML-dLDS robustness, we add *i.i.d.* Gaussian observation noise whose standard deviation σ we increase, and rerun the same pipeline. Circuit and delay recovery show a phase transition as the noise grows, with recovery holding for $\sigma \leq 0.1$, declining through $\sigma{=}0.2$, and collapsing by $\sigma{=}0.3$ (Fig. 12).

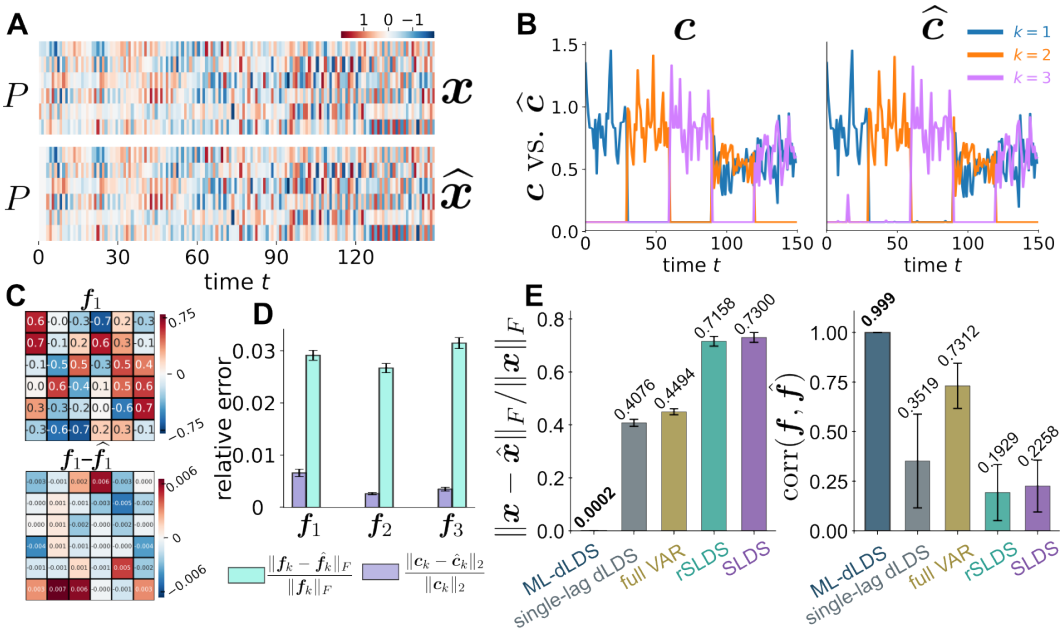

Figure 2: **Circuit, lag, and coefficient recovery on the first synthetic dataset.** **(A)** Latent $\boldsymbol{x}$ (top) and its reconstruction $\hat{\boldsymbol{x}}$ (bottom); relative ℓ_2 error 2×10^{-4}. **(B)** True (left) and recovered (right) coefficients $\boldsymbol{c}_{k,t}$. **(C)** Circuit $\boldsymbol{f}_1$ (top) and residual $\boldsymbol{f}_1 - \hat{\boldsymbol{f}}_1$ (bottom), shown at the 10^{-3} scale. **(D)** Per-circuit relative ℓ_2 error for the circuits, $\|\boldsymbol{f}_k - \hat{\boldsymbol{f}}_k\|_F / \|\boldsymbol{f}_k\|_F$, and for the coefficients, $\|\boldsymbol{c}_k - \hat{\boldsymbol{c}}_k\|_2 / \|\boldsymbol{c}_k\|_2$. **(E)** Baselines over multiple seeds: reconstruction error (left, lower is better) and circuit correlation $\mathrm{corr}(\boldsymbol{f}, \hat{\boldsymbol{f}})$ (right, higher is better); error bars are standard deviation across seeds.

4.2 ML-dLDS SELECTS SPARSE DELAYS UNDER AMBIGUITY

For the second dataset we generated P=6 dimensions, K=3 orthogonal circuits, Δ=4 delays, and T=500 time points (additional data details are in App. F). As in the first dataset, the coefficients $\boldsymbol{c}$ stay positive and fluctuate fast under a schedule that activates circuits in mixtures. Here circuit $\boldsymbol{f}_1$ acts at two delays, while $\boldsymbol{f}_2$ and $\boldsymbol{f}_3$ act at one delay each, which challenges the task due to possible ambiguity between its candidate delays.

When we apply ML-dLDS to this data, it recovers the delay support, the circuits ($\mathrm{corr}(\boldsymbol{f}, \hat{\boldsymbol{f}})$=0.960), and the coefficients (Fig. 3A-C,E). We then compare ML-dLDS against the baselines, where it outperforms single-lag dLDS, full VAR, rSLDS, and SLDS on both reconstruction accuracy and circuit-to-circuit correlation (Fig. 3D,F). We further confirm these gaps by running both ML-dLDS and each baseline across 200 seeds and comparing them with a paired Wilcoxon signed-rank test. ML-dLDS exceeds all four on both metrics, with all $p < 10^{-15}$ (Tab. 2).

Next, to test the importance of each model component, we ablate them one at a time (Fig. 3G-I). Removing the lag penalty on $\boldsymbol{S}$, or keeping only its ℓ_2 term, drops lag-support recovery, which emphasizes the importance of the ℓ_1 term (Fig. 3G). Fixing $\boldsymbol{c}$ in time, or reducing the model to a single circuit (with or without delays), degrades reconstruction and circuit correlation (Fig. 3H,I).

We also test recovery stability to hyperparameters and find that it holds across penalty weights and maximum allowed lag (Fig. 8, Fig. 6 top, Fig. 7), as well as under increasing observation noise (Fig. 6 bottom) and across random seeds (App. D.2). In addition, we test how well the recovered lags generalize by predicting held-out time. We hold out the later part of each synthetic recording, re-solve only the coefficients $\boldsymbol{c}$ on it, and predict each next step from past states. The multi-lag model predicts the next step more accurately than the same model restricted to a single lag ($\Delta = 1$) on both synthetic datasets (Fig. 11).

4.3 ML-dLDS REVEALS MULTI-LAG INTERAREAL DYNAMICS IN MOUSE CORTEX

Next, we apply ML-dLDS to Neuropixels recordings from a mouse performing a visual discrimination task (Steinmetz et al., 2019), which we extract through the DANDI Archive (dandiset 000017).

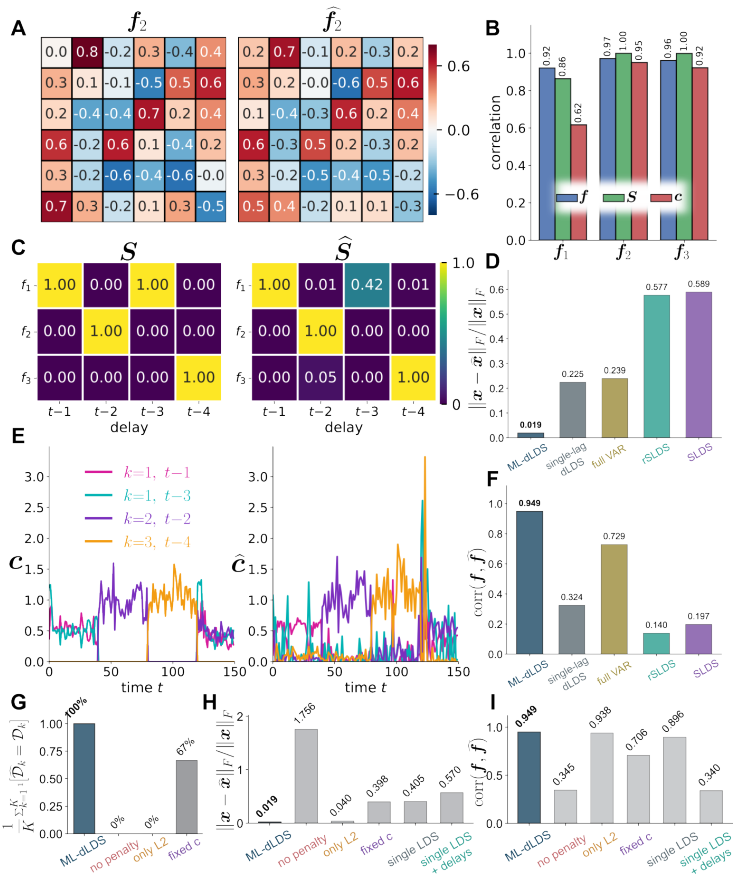

Figure 3: **Results on the second synthetic dataset.** **(A)** Example circuit ($\boldsymbol{f}_2$) true vs. recovered. **(B)** Per-circuit correlation for $\boldsymbol{f}$, $\boldsymbol{S}$, and $\boldsymbol{c}$. **(C)** $\boldsymbol{S}$ true and recovered; $\boldsymbol{f}_1$ uses $t-1$ and $t-3$. **(D)** Reconstruction error against baselines. **(E)** $\boldsymbol{c}_{k,t}$ true and recovered. **(F)** Circuit correlation against baselines. **(G)** Lag-support match across penalty ablations; no penalty or L_2 alone gives 0%, ML-dLDS 100%. **(H)** Reconstruction error across ablations. **(I)** Circuit correlation across ablations.

In the task, the mouse reports the side of a visual stimulus, which provides trial-aligned behavioral variables (choice, contrast, feedback). We analyze an example session with $N = 326$ simultaneously recorded units across three areas, primary visual cortex (VISp), lateral posterior thalamus (LP), and secondary motor cortex (MOs), and estimate firing rates in 50 ms time bins (Fig. 4B).

ML-dLDS identifies circuits whose lag profiles $\boldsymbol{S}$ spread across delays $\delta = 1$ to 6, so the population dynamics combine interactions at several communication delays (Fig. 4F, Fig. 13). When we project the circuits to the observed space and group by brain area, interactions are strongest within and outgoing from LP at delay $\delta = 1$ and re-emerge at $\delta = 4$ (Fig. 4D), consistent with delayed interareal communication in the LP (e.g. (Liu et al., 2022)).

To test whether this structure carries behavior, we decode task variables from the latent trajectories. Latent states separate by the animal's choice after stimulus onset (Fig. 4E), and choice is decodable well above chance in all three areas (Fig. 4G). Time-resolved decoding shows choice information at chance before the stimulus and rising sharply after onset (Fig. 4H), so the recovered dynamics track the emerging decision rather than a static stimulus response.

We further ask whether the recovered delays help predict future activity: we hold out trials and predict each neural state one step ahead from past states, re-solving only the coefficients $\boldsymbol{c}$ on the held-out trials. Held-out one-step prediction improves with multiple delays, reaching correlation 0.70 ± 0.003 versus 0.63 ± 0.006 for a single-lag restriction of the same model (mean $\pm$ SEM across five folds, paired t-test $p < 0.001$) (Fig. 14).

Finally, we compare ML-dLDS against baselines (Fig. 4I,J), where it reaches better reconstruction than existing methods and matches the full VAR while using about $200\times$ fewer parameters (Tab. 3). Crucially, unlike the full VAR, which exposes only a single dense transition, ML-dLDS returns interpretable shared circuits $\boldsymbol{f}_k$ with their delay profiles $\boldsymbol{S}$ and per-circuit coefficients $\boldsymbol{c}_t$.

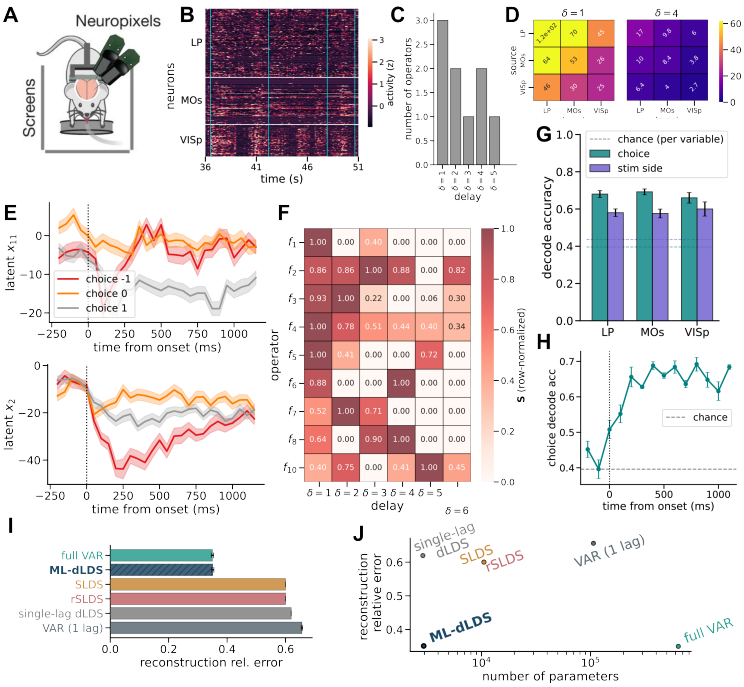

Figure 4: **ML-dLDS finds multi-delay latent dynamics from Neuropixels recordings.** LP, MOs, VISp during a visual decision task. **(A)** Recording setup. **(B)** z-scored spiking. **(C)** Circuits per delay δ. **(D)** Cross-area strength at δ=1 and δ=4; longer delays weaker. **(E)** Latent coordinates, aligned to onset; separate by choice. **(F)** Row-normalized $\boldsymbol{S}$; several circuits load multiple delays. **(G)** Decoding per area, above chance. **(H)** Choice decoding rises after onset. **(I)** Reconstruction error; ML-dLDS matches full VAR, beats others. **(J)** Error vs parameter count; ML-dLDS matches VAR with fewer parameters.

4.4 ML-dLDS GENERALIZES BEYOND NEUROSCIENCE

While neural dynamics motivate ML-dLDS, its multi-lag structure applies wherever coupled signals interact across delays. To show this generality, we apply it to COVID-19 case data from Our World in Data (Hasell et al., 2020), where the recovered circuits align with known pandemic phases (Fig. 5A). The data covers 49 countries in weekly bins from 2020 to 2023. For each country we take new-case counts, smooth them, and compute the weekly log-growth.

We fit ML-dLDS with $K = 10$ circuits and $\Delta = 8$ weekly delays, so each circuit captures how countries drive one another at a delay of up to eight weeks. We found circuits that act at several delays, and their scaling $\boldsymbol{S}$ spreads across lags rather than collapsing to one (Fig. 5B). We further found that these circuits are active in phases that align with real trends in COVID-19, where one circuit peaks during the first 2020 waves, another during the Alpha and Delta period, and a third during the Omicron surge and the 2023 sublineage waves (Fig. 5D).

The components identified by ML-dLDS further prefer short delays that act via the dynamic coefficients during the synchronized 2021 to 2022 waves, while coefficients linked to longer delays emerge in the desynchronized 2023 phase (Fig. 5E). We then tested whether the circuits overlap in time, and found that a single circuit dominates each week far more than a circular-shift null allows ($p < 0.001$ via one-sided shuffle test), so ML-dLDS partitions the timeline into distinct temporal regimes (App. I).

Removing any single circuit raises the reconstruction error, and each circuit proves necessary within its own active phase (Fig. 5C). ML-dLDS reconstructs the data with relative error 0.03 (Fig. 5F). Compared to baselines, including single-lag VAR, single-lag dLDS, and the switching models rSLDS and SLDS, ML-dLDS reaches the lowest error (Fig. 5F). We could not compare to full VAR here, since at $\Delta = 8$ it fits a separate $P \times P$ matrix per lag, giving more unknowns than the 199 time points can determine.

5 DISCUSSION, LIMITATIONS, AND FUTURE STEPS

We present ML-dLDS, a data-driven model of time-varying neural population dynamics whose parallel processes act across multiple co-occurring lags. Such delays are intrinsic to how the brain works, as neural communication proceeds at lags set by pathway, region, and task (Semedo et al., 2022; Tauste Campo et al., 2015; Moon et al., 2022). This variation goes largely unaddressed in

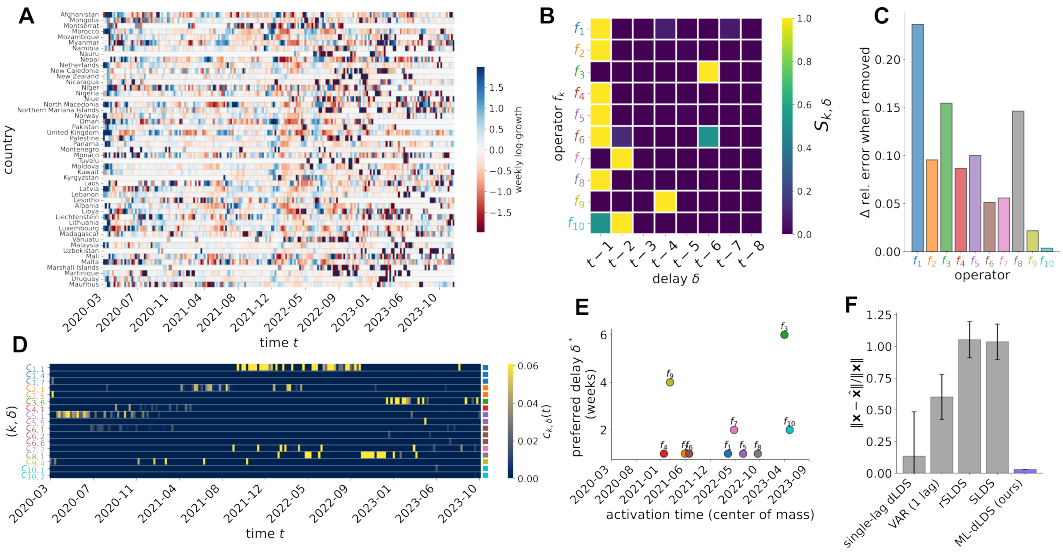

Figure 5: **ML-dLDS reveals interpretable multi-lag structure in COVID-19 case dynamics across countries.** **(A)** Weekly log-growth of new cases, 49 countries, 2020 to 2023 (input $\boldsymbol{x}$). **(B)** Row-normalized $\boldsymbol{S}$, circuit by delay; several circuits load multiple delays. **(C)** Per-circuit ablation; every circuit raises reconstruction error when removed. **(D)** Coefficients $c_{k,\delta}(t)$ over time. **(E)** Preferred delay δ^* against activation time. **(F)** Reconstruction error against baselines (error bars, standard deviation across time). ML-dLDS reaches the lowest error; rSLDS and SLDS reach higher error, and full VAR cannot run at $\Delta = 8$ on 199 time points.

earlier computational work, where a single fixed lag merges a fast local interaction and a slow long-range one into one operator. ML-dLDS instead learns each lag explicitly along with its corresponding interpretable dynamics, so every learnt component reads as one specific population interaction at the delays it acts on.

ML-dLDS thereby opens a question earlier models could not ask: on what timescale does a given interaction play out, which population drives which, and how does that coupling evolve? We put this question to multi-region mouse cortex data from Steinmetz et al. (2019), where the fitted operators answer it directly from the dynamics rather than from a separate delay estimate.

The value of recovered hidden interactions along with their time varying properties across lags is not limited to the brain, since any system of coupled signals can carry interactions at more than one delay. To test whether this holds outside neuroscience, we also turned ML-dLDS on COVID-19 case counts across countries, where one nation's trajectory influences another over a range of delays. There, it found circuits that act at their own delays and whose activations track known pandemic phases (Fig. 5).

Limitations and Future Work. ML-dLDS rests on several assumptions that bound the complexity it can capture and mark directions for future work. First, ML-dLDS treats each interaction as a linear map, which aids interpretability but leaves nonlinear dynamics out of reach, a gap future work could close through kernelization. Second, the gain $\boldsymbol{S}$ sets one strength per circuit and lag, shared across all latent dimensions, so a circuit drives every latent at that lag with the same gain. This simplicity aids both interpretability and fitting robustness, though allowing the lag to vary per latent could be an exciting direction for future extensions. One more limitation occurs when two candidate lags are nearly indistinguishable in the data, which the weighted lag-sparsity penalty handles by favoring the shorter, more parsimonious lag.

One more interesting direction for future neuroscience work folds spatial information into the inference as a regularizer, so anatomically adjacent latents share lag and gain structure during the fit rather than after it. A related direction constrains each circuit to a brain-area block pattern, with off-block entries masked (e.g., as in (Mudrik et al., 2024b)), so circuits capture within-area or cross-area interactions by construction.

THE USE OF LARGE LANGUAGE MODELS (LLMS)

We used generative AI tools to correct grammar and typos, to provide sentence-level phrasing suggestions, to assist with code debugging, and to help with LaTeX formatting. We did not use generative AI tools to generate research ideas, to design the method, to derive results, or to retrieve and cite related work. All scientific content, analyses, and conclusions are the authors' own. We reviewed and verified every AI-assisted output, and we take full responsibility for the final content of this work.

REFERENCES

Larry F Abbott and Sacha B Nelson. Synaptic plasticity: taming the beast. *Nature neuroscience*, 3 (11):1178–1183, 2000.

Yenho Chen, Noga Mudrik, Kyle A Johnsen, Sankaraleengam Alagapan, Adam S Charles, and Christopher J Rozell. Probabilistic decomposed linear dynamical systems for robust discovery of latent neural dynamics. *Advances in Neural Information Processing Systems*, 37:104443–104470, 2024.

David L Donoho and Michael Elad. Optimally sparse representation in general (nonorthogonal) dictionaries via l1 minimization. *Proceedings of the National Academy of Sciences*, 100(5):2197–2202, 2003.

Joshua Glaser, Matthew Whiteway, John P Cunningham, Liam Paninski, and Scott Linderman. Recurrent switching dynamical systems models for multiple interacting neural populations. *Advances in neural information processing systems*, 33:14867–14878, 2020.

Evren Gokcen, Anna I Jasper, João D Semedo, Amin Zandvakili, Adam Kohn, Christian K Machens, and Byron M Yu. Disentangling the flow of signals between populations of neurons. *Nature Computational Science*, 2(8):512–525, 2022.

Evren Gokcen, Anna Jasper, Alison Xu, Adam Kohn, Christian K Machens, and Byron M Yu. Uncovering motifs of concurrent signaling across multiple neuronal populations. *Advances in Neural Information Processing Systems*, 36, 2024.

Gene H Golub and Charles F Van Loan. *Matrix computations*, volume 3. JHU press, 2013.

Kameron Decker Harris, Aleksandr Aravkin, Rajesh Rao, and Bingni Wen Brunton. Time-varying autoregression with low-rank tensors. *SIAM Journal on Applied Dynamical Systems*, 20(4):2335–2358, 2021.

Joe Hasell, Edouard Mathieu, Diana Beltekian, Bobbie Macdonald, Charlie Giattino, Esteban Ortiz-Ospina, Max Roser, and Hannah Ritchie. Covid-19 testing: A cross-country database. 2020.

Orren Karniol-Tambour, David M. Zoltowski, E. Mika Diamanti, Lucas Pinto, Carlos D Brody, David W. Tank, and Jonathan W. Pillow. Modeling state-dependent communication between brain regions with switching nonlinear dynamical systems. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=WQwV7Y8qwa.

Hyun Dong Lee, Andrew Warrington, Joshua Glaser, and Scott Linderman. Switching autoregressive low-rank tensor models. *Advances in Neural Information Processing Systems*, 36:57976–58010, 2023.

Weihan Li, Chengrui Li, Yule Wang, and Anqi Wu. Multi-region markovian gaussian process: An efficient method to discover directional communications across multiple brain regions. *Proceedings of machine learning research*, 235:28112, 2024a.

Weihan Li, Yule Wang, Chengrui Li, and Anqi Wu. Learning time-varying multi-region brain communications via scalable markovian gaussian processes. *arXiv preprint arXiv:2407.00397*, 2024b.

Scott W Linderman, Andrew C Miller, Ryan P Adams, David M Blei, Liam Paninski, and Matthew J Johnson. Recurrent switching linear dynamical systems. *arXiv preprint arXiv:1610.08466*, 2016.

Denghui Liu, Shouhao Li, Liqing Ren, Xiaoyuan Li, and Zhenlong Wang. The superior colliculus/lateral posterior thalamic nuclei in mice rapidly transmit fear visual information through the theta frequency band. *Neuroscience*, 496:230–240, 2022.

Eve Marder. Neuromodulation of neuronal circuits: back to the future. *Neuron*, 76(1):1–11, 2012.

Joon-Young Moon, Kathrin Müsch, Charles E Schroeder, Taufik A Valiante, and Christopher J Honey. Inter-regional delays fluctuate in the human cerebral cortex. *bioRxiv*, pp. 2022–06, 2022.

Noga Mudrik, Yenho Chen, Eva Yezerets, Christopher J Rozell, and Adam S Charles. Decomposed linear dynamical systems (dlds) for learning the latent components of neural dynamics. *Journal of Machine Learning Research*, 25(59):1–44, 2024a.

Noga Mudrik, Ryan Ly, Oliver Ruebel, and Adam S Charles. Creimbo: Cross ensemble interactions in multi-view brain observations. *arXiv preprint arXiv:2405.17395*, 2024b.

Chethan Pandarinath, Daniel J O'Shea, Jasmine Collins, Rafal Jozefowicz, Sergey D Stavisky, Jonathan C Kao, Eric M Trautmann, Matthew T Kaufman, Stephen I Ryu, Leigh R Hochberg, et al. Inferring single-trial neural population dynamics using sequential auto-encoders. *Nature methods*, 15(10):805–815, 2018.

Matthew G Perich, Charlotte Arlt, Sofia Soares, Siyan Zhou, Manuel Beiran, Aaron S Andalman, Tyler Benster, Megan E Young, Clayton P Mosher, Juri Minxha, et al. Inferring brain-wide interactions using data-constrained recurrent neural network models. *Neuron*, 2021.

Jonathan W Pillow, Jonathon Shlens, Liam Paninski, Alexander Sher, Alan M Litke, EJ Chichilnisky, and Eero P Simoncelli. Spatio-temporal correlations and visual signalling in a complete neuronal population. *Nature*, 454(7207):995–999, 2008.

João D Semedo, Anna I Jasper, Amin Zandvakili, Aravind Krishna, Amir Aschner, Christian K Machens, Adam Kohn, and Byron M Yu. Feedforward and feedback interactions between visual cortical areas use different population activity patterns. *Nature communications*, 13(1):1099, 2022.

Nicholas A Steinmetz, Peter Zatka-Haas, Matteo Carandini, and Kenneth D Harris. Distributed coding of choice, action and engagement across the mouse brain. *Nature*, 576(7786):266–273, 2019.

James H Stock and Mark W Watson. Vector autoregressions. *Journal of Economic perspectives*, 15 (4):101–115, 2001.

Adrià Tauste Campo, Marina Martinez-Garcia, Verónica Nácher, Rogelio Luna, Ranulfo Romo, and Gustavo Deco. Task-driven intra-and interarea communications in primate cerebral cortex. *Proceedings of the National Academy of Sciences*, 112(15):4761–4766, 2015.

Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, pp. 267–288, 1996.

Joel A Tropp. Greed is good: Algorithmic results for sparse approximation. *IEEE Transactions on Information theory*, 50(10):2231–2242, 2004.

E. van den Berg and M. P. Friedlander. SPGL1: A solver for large-scale sparse reconstruction, December 2019. https://friedlander.io/spgl1.

Ewout Van Den Berg and Michael P Friedlander. Probing the pareto frontier for basis pursuit solutions. *Siam journal on scientific computing*, 31(2):890–912, 2009.

Alexander Waibel, Toshiyuki Hanazawa, Geoffrey Hinton, Kiyohiro Shikano, and Kevin J Lang. Phoneme recognition using time-delay neural networks. *IEEE transactions on acoustics, speech, and signal processing*, 37(3):328–339, 1989.

Joel Ye and Chethan Pandarinath. Representation learning for neural population activity with neural data transformers, 2021. URL `https://arxiv.org/abs/2108.01210`.

Qunxi Zhu, Yao Guo, and Wei Lin. Neural delay differential equations. *arXiv preprint arXiv:2102.10801*, 2021.

Appendix

A WARM-START INITIALIZATION

A poor circuit estimate can degrade the lag assignment, so we initialize both from the data before the loop. Over a segment where a single circuit dominates, the dynamics reduce to one linear map at one lag, and a single-map least-squares fit attains high R^2. We scan the recording for these locally simple segments and read each one's lag from the fit that best explains it. We cluster the detected maps into K groups and initialize the circuits from the orthogonalized centroids.

Clustering discovers the circuits and their lags from the observations alone, and the alternating loop refines them. The sparsity on $\boldsymbol{S}$ does not discover lags, it enforces few lags per circuit once the loop runs. The detection assumes locally simple segments exist, which our synthetic data provide and which recordings may lack, in which case the alternating loop still refines the fit from a rougher start.

Algorithm 1 ML-dLDS fitting

Require: observations $\boldsymbol{Y}$, loading $\boldsymbol{A}$, number of circuits K, max lag Δ, weights $\lambda_{\text{dyn}}, \lambda_c, \lambda_1, \lambda_2, \lambda_3, \mu$
Ensure: latents $\boldsymbol{x}$, activations $\boldsymbol{c}$, scaling $\boldsymbol{S}$, circuits $\{\boldsymbol{f}_k\}$

initialize $\{\boldsymbol{f}_k\}$ from detected segments of locally simple dynamics ▷ warm start, App. A
repeat
  for each time t **do**
    update $\boldsymbol{x}_t, \boldsymbol{c}_t$ jointly by equation 6 ▷ co-inference
  end for
  update $\boldsymbol{S}$ by equation 7 ▷ lag step
  normalize each row of $\boldsymbol{S}$ to unit maximum, absorb the scale into $\boldsymbol{c}$
  for $k = 1, \ldots, K$ **do**
    update $\boldsymbol{f}_k$ by equation 8 ▷ circuit step, Gauss-Seidel
  end for
until schedule complete (App. B)

B OPTIMIZATION SCHEDULE

We run the fit for a fixed budget of three outer passes with no early stopping. Each outer pass runs three inner iterations of the co-inference and $\boldsymbol{S}$ steps before a single circuit update, since a fully converged inner solve wastes effort while the circuits are still rough. After each $\boldsymbol{S}$ step we normalize every row to unit maximum and absorb the scale into $\boldsymbol{c}$, which fixes the split between $\boldsymbol{S}$ and $\boldsymbol{c}$. The loading $\boldsymbol{A}$ is initialized with random orthonormal columns and re-estimated each outer pass by a ridge least-squares step against the current latent, $\boldsymbol{A} = (\boldsymbol{Y}\boldsymbol{X}^\top)(\boldsymbol{X}\boldsymbol{X}^\top + \lambda_A \boldsymbol{I})^{-1}$, followed by a polar projection to orthonormal columns that fixes the gauge, where λ_A is the only $\boldsymbol{A}$-step hyperparameter.

C BASELINES

We compare ML-dLDS against four baselines, each relaxing one of its assumptions. *Single-lag dLDS* (Mudrik et al., 2024a) forces Δ=1, isolating the effect of learning delays. *Full VAR* (Stock & Watson, 2001) fits a separate circuit per lag with no sharing and no sparsity, at far more parameters; single-lag VAR is its one-lag version. *rSLDS* (Linderman et al., 2016) and *SLDS* represent the switching family, one active mode at a time. *MRM-GP* (Li et al., 2024a) represents the region-based family, recovering directional phase delays between regions rather than a shared operator set.[1] We ran it on the same binned session with the real area labels (VISp, LP, MOs). MRM-GP is built to model a pair of regions at a time. Hence, we designed the pairwise setup to align best

[1] Code from `https://github.com/WeihanLikk/MRM-GP`.

Table 1: Recovery on identifiable synthetic data (dataset 1, scalar $\boldsymbol{S}$). $\boldsymbol{f}$ correlation and delay support are higher-is-better, reconstruction relative error is lower-is-better. ML-dLDS recovers the operators and the delay assignment exactly. Single-lag dLDS keeps only the lag-1 operator, full VAR uses independent per-lag operators, and rSLDS and SLDS switch among discrete regimes, so none exposes a shared delay-indexed operator set. The last two columns report paired Wilcoxon signed-rank p-values across n=100 seeds for ML-dLDS versus each baseline, on circuit correlation (p_{corr}) and on reconstruction error (p_{rec}). ML-dLDS exceeds every baseline on both metrics on essentially every seed.

model	$\mathrm{corr}(\boldsymbol{f}, \widehat{\boldsymbol{f}})$	delay support	recon. rel. err.	params	p_{corr}	p_{rec}
ML-dLDS (ours)	$\mathbf{1.000}$	$\mathbf{100}\%$	$\mathbf{2{\times}10^{-4}}$	117	–	–
single-lag dLDS	0.352	33%	0.408	111	$3.9{\times}10^{-18}$	$3.9{\times}10^{-18}$
full VAR	0.731	–	0.449	108	$2.2{\times}10^{-16}$	$3.9{\times}10^{-18}$
rSLDS	0.193	–	0.716	–	$4.0{\times}10^{-18}$	$3.9{\times}10^{-18}$
SLDS	0.226	–	0.730	–	$3.9{\times}10^{-18}$	$3.9{\times}10^{-18}$

Table 2: Delay-sparse recovery (dataset 2, Δ=4, operator $\boldsymbol{f}_1$ at two delays). Top block, baselines. Bottom block, fitting ablations of ML-dLDS, each changing one component. Support is undefined for models without a delay-indexed operator set and for the single-operator ablations. The last two columns report paired Wilcoxon signed-rank p-values across n=200 seeds for ML-dLDS versus each baseline, on circuit correlation (p_{corr}) and on reconstruction error (p_{rec}). ML-dLDS exceeds every baseline on both metrics on essentially every seed. The ablations are reported at a single seed.

method	$\mathrm{corr}(\boldsymbol{f}, \widehat{\boldsymbol{f}})$	delay support	recon. rel. err.	p_{corr}	p_{rec}
ML-dLDS (ours)	$\mathbf{0.949}$	$\mathbf{100}\%$	$\mathbf{0.019}$	–	–
single-lag dLDS	0.324	–	0.225	$1.5{\times}10^{-34}$	$1.4{\times}10^{-34}$
full VAR	0.729	–	0.239	$5.3{\times}10^{-32}$	$1.4{\times}10^{-34}$
rSLDS	0.140	–	0.577	$1.4{\times}10^{-34}$	$1.4{\times}10^{-34}$
SLDS	0.197	–	0.589	$1.5{\times}10^{-34}$	$1.4{\times}10^{-34}$
no penalty	0.345	0%	1.756	–	–
only ℓ_2	0.938	0%	0.040	–	–
fixed $\boldsymbol{c}$	0.706	67%	0.398	–	–
single LDS	0.896	–	0.405	–	–
single LDS + delays	0.340	–	0.570	–	–

with their method, fitting all three possible region combination pairs (VISp-LP, VISp-MOs, LP-MOs), to suit its two-region form and give the fairest comparison we could. We report the mean reconstruction relative error in observed space, scored as for the other baselines. Because MRM-GP recovers region-pair phase delays rather than a delay-indexed operator dictionary, it exposes neither operators nor a delay assignment to score against the ground truth. Notably, MRM-GP requires data partitioned into regions, so we apply it only to the neural session, which has real brain areas labels. We note that we focus on baselines that produce interpretable interaction components, which is at the core of this paper's goal of understanding how the dynamics evolve over time, and which more complex deep-learning methods, that recover often uninterpretable latent dynamics (e.g. LFADS (Pandarinath et al., 2018)), do not provide, so a comparison would not be meaningful.

D METRICS AND SCORING

All methods use seed 0, data scaled to unit standard deviation with no other preprocessing, and one permutation-aware scoring module. Circuits are matched to the ground truth by the best permutation before any metric. Support match is the fraction of circuits whose recovered lag set equals the true set, where the recovered set collects lags with row-normalized $\boldsymbol{S}$ above 0.3. Circuit correlation is the mean over K of $|\mathrm{corr}|$ between matched circuits, and reconstruction error is $\|\boldsymbol{x} - \widehat{\boldsymbol{x}}\|_F / \|\boldsymbol{x}\|_F$.

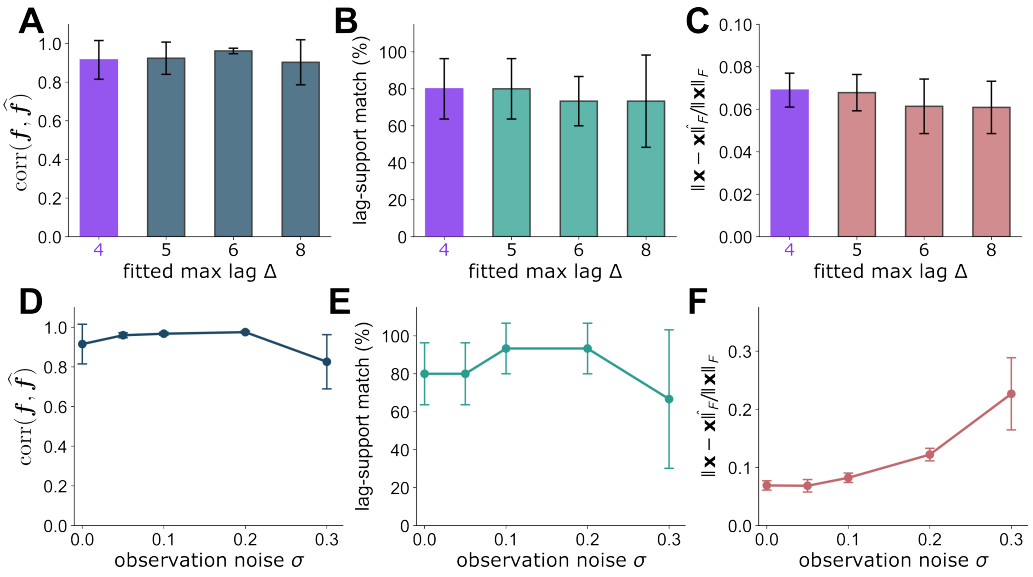

Figure 6: **Delay recovery on synthetic 2 stays stable across a misspecified maximum lag and observation noise.** Recovery is scored as circuit correlation $\mathrm{corr}(\boldsymbol{f}, \widehat{\boldsymbol{f}})$, lag-support match, and reconstruction relative error $\|\boldsymbol{x} - \widehat{\boldsymbol{x}}\|_F / \|\boldsymbol{x}\|_F$. **(A–C)** Metrics against the fitted maximum lag Δ, where the true value $\Delta = 4$ appears in purple, and recovery tolerates one extra lag before support degrades. **(D–F)** Metrics against observation noise σ, where recovery holds through $\sigma = 0.2$ and declines by $\sigma = 0.3$. Bars and points show the mean and error bars the standard deviation across 5 seeds. The detector finds no clean single-circuit window at $\sigma = 0.5$, so that level carries no recovery (Fig. 12).

D.1 Robustness of delay recovery on synthetic 2

We tested whether the delay recovery on synthetic 2 depends on the exact hyperparameters, the assumed maximum lag, or the noise level. Across every check the recovery stays stable, and the setting used in the main text sits inside the stable region.

We first swept the lag-sparsity weight λ_1 and the stability weight λ_3 over a grid, holding the orthogonality weight at $\lambda_2 = 0$ as in the main text (Fig. 8). Lag-support recovery reaches 100% over a broad interior region that contains the main-text setting $(\lambda_1, \lambda_3) = (0.2,\ 2.0)$. Recovery breaks only when λ_1 grows large enough to over-sparsify the lag profiles. The $\lambda_3 = 0$ column is degenerate, since without the stability term the lag-scaling solve is ill-conditioned and its recovery is unreliable.

We then varied the fitted maximum lag Δ and the observation noise σ (Fig. 6). Recovery tolerates one lag beyond the true $\Delta = 4$ and degrades once the extra lags dilute the support. Recovery holds through $\sigma = 0.2$ and declines by $\sigma = 0.3$, consistent with the noise behavior on synthetic 1 (Fig. 12).

D.2 Multi-seed robustness on synthetic data

To confirm that recovery does not rest on a single favorable draw, we regenerated each synthetic dataset across many independent seeds and refit with fixed hyperparameters and no per-seed tuning. We report the mean and the standard deviation across seeds, so the error bars measure run-to-run variability rather than uncertainty within one fit. Alongside the binary support match of App. D, we report a graded per-circuit score, the correlation $\mathrm{corr}(\boldsymbol{S}_k, \hat{\boldsymbol{S}}_k)$ between the true and recovered lag profiles, which credits partial recovery that the exact-set match discards.

On synthetic 1, recovery held high and stable across 100 seeds, with $\mathrm{corr}(\boldsymbol{f}, \hat{\boldsymbol{f}}) = 0.96 \pm 0.08$, lag support $97\% \pm 10\%$, and reconstruction relative error $\|\boldsymbol{x} - \hat{\boldsymbol{x}}\| / \|\boldsymbol{x}\|$ at 0.075 ± 0.053. This stability shows that the single-seed recovery in Fig. 2 reflects the method and not one lucky initialization.

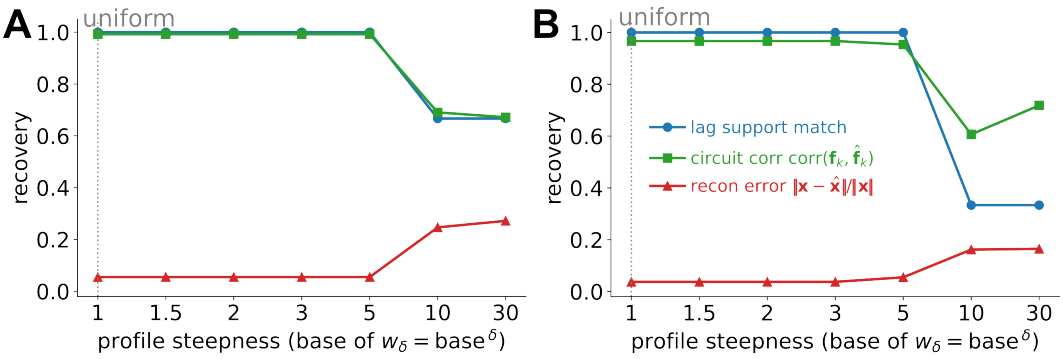

Figure 7: **Delay recovery stays robust to the lag-weight profile w_δ and degrades only under an extreme profile. (A)** On synthetic 1, where each operator uses a single delay, the lag support match (blue), circuit correlation $\mathrm{corr}(\boldsymbol{f}_k, \hat{\boldsymbol{f}}_k)$ (green), and reconstruction error $|\boldsymbol{x} - \hat{\boldsymbol{x}}|/|\boldsymbol{x}|$ (red) stay flat for every profile up to $\mathrm{base} = 5$ and drop once the profile grows steeper, where the per-lag weight follows $w_\delta = \mathrm{base}^{\,\delta}$ and $\mathrm{base} = 1$ gives the uniform profile (plain elastic net). **(B)** On synthetic 2, where operator $\boldsymbol{f}_1$ uses two delays ($t-1$ and $t-3$), the same three metrics hold their plateau up to $\mathrm{base} = 5$, and a steeper profile collapses the two-delay support of $\boldsymbol{f}_1$ and raises the reconstruction error.

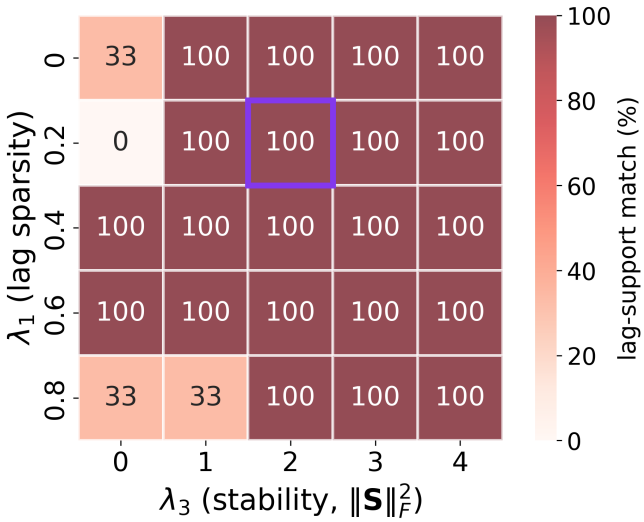

Figure 8: **Lag-support recovery on synthetic 2 is stable across the lag-sparsity and stability weights.** Each cell reports the lag-support match for one (λ_1, λ_3) pair at seed 0, with the orthogonality weight held at $\lambda_2 = 0$. The main-text setting $(\lambda_1, \lambda_3) = (0.2,\ 2.0)$ is outlined in purple and sits inside a broad 100% plateau. The $\lambda_3 = 0$ column is degenerate, since without the stability term the lag-scaling solve is ill-conditioned, so its values reflect that instability rather than a true sparsity effect.

On synthetic 2, across 200 seeds ML-dLDS reached $\mathrm{corr}(\boldsymbol{f}, \hat{\boldsymbol{f}}) = 0.92 \pm 0.10$, reconstruction error 0.071 ± 0.027, and mean lag support $76\% \pm 26\%$ (Fig. 9). The graded lag-profile score localizes the difficulty to the two-lag circuit. The single-lag circuits recover their profile almost perfectly, at $\mathrm{corr}(\boldsymbol{S}_k, \hat{\boldsymbol{S}}_k) = 0.96$ for $\boldsymbol{f}_2$ and 0.95 for $\boldsymbol{f}_3$, while the two-lag circuit $\boldsymbol{f}_1$ reaches 0.78. The elastic net recovers both lags of $\boldsymbol{f}_1$ on 63% of seeds, and the residual failures come from the ℓ_1 lag-sparsity collapsing the two lags to one. This term stays necessary for delay selection, since removing it drops lag support to zero (Fig. 3G), yet it sometimes over-sparsifies a genuine second lag. The coefficient traces keep the two lags of $\boldsymbol{f}_1$ separable and the effective design-column correlation stays moderate rather than degenerate, so the shortfall is a limit of the sparsity penalty and not an identifiability limit.

E Synthetic 1, Additional Information

The first dataset has P=6, K=3, Δ=3, T=900, one lag per circuit, orthogonal circuits. For the baselines in that experiment, single-lag dLDS uses solo-detect init (window 25, stride 8, threshold

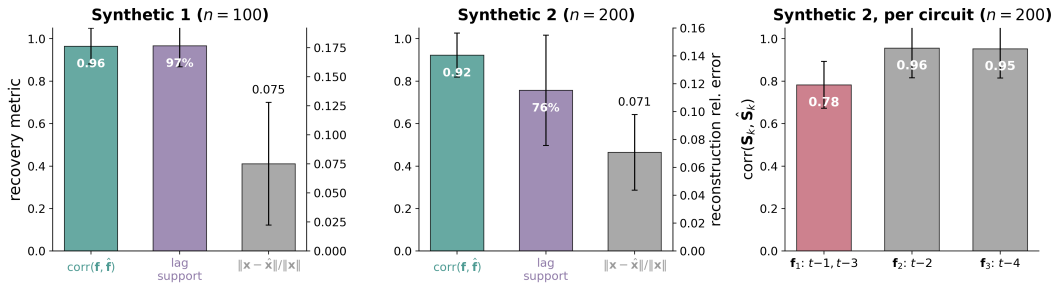

Figure 9: **Multi-seed recovery on synthetic data, error bars are the standard deviation across seeds. (Left)** Synthetic 1 over 100 seeds, with circuit correlation $\mathrm{corr}(\boldsymbol{f}, \hat{\boldsymbol{f}})$, lag support, and reconstruction relative error $\|\boldsymbol{x} - \hat{\boldsymbol{x}}\|/\|\boldsymbol{x}\|$. **(Middle)** Synthetic 2 over 200 seeds, same metrics. **(Right)** Synthetic 2 graded lag-profile recovery per circuit, $\mathrm{corr}(\boldsymbol{S}_k, \hat{\boldsymbol{S}}_k)$. The single-lag circuits $\boldsymbol{f}_2$ and $\boldsymbol{f}_3$ recover their profile almost perfectly, and the two-lag circuit $\boldsymbol{f}_1$, at $t{-}1$ and $t{-}3$, reaches 0.78. The middle lag support is the binary support match of App. D, while the right panel is the graded lag-profile correlation, so the two panels measure delay recovery differently.

0.94), three outer by three inner steps, $\boldsymbol{f}$-step ridge 10^{-6}, and has $KP^2{+}K{=}111$ parameters. Full VAR solves $\boldsymbol{B} = \boldsymbol{Y}\boldsymbol{Z}^\top(\boldsymbol{Z}\boldsymbol{Z}^\top + \lambda \boldsymbol{I})^{-1}$ with $\lambda{=}10^{-4}$, $\Delta P^2{=}108$ parameters, closed form. rSLDS and SLDS use the `ssm` library (v0.0.1), $D{=}6$, frozen identity emission, Laplace EM, 25 iterations, recurrent and standard transitions.

F Synthetic 2, Additional Information

The second dataset has $P{=}6$, $K{=}3$, $\Delta{=}4$, $T{=}500$, and true support $\mathcal{D}_1{=}\{t{-}1, t{-}3\}$, $\mathcal{D}_2{=}\{t{-}2\}$, $\mathcal{D}_3{=}\{t{-}4\}$.

The four baselines in this experiment match their synthetic 1 form, with $\Delta{=}4$ and $T{=}500$. *Single-lag dLDS* forces $\Delta{=}1$ and uses the synthetic 2 solo-detect defaults (window 22, stride 10, threshold 0.88); every other setting matches ML-dLDS. *Full VAR* sums over four lags, $\boldsymbol{x}_t = \sum_{\delta=1}^{4} \boldsymbol{A}_\delta \boldsymbol{x}_{t-\delta}$, with the same ridge $\lambda{=}10^{-4}$ and closed-form solve, giving $\Delta P^2{=}144$ parameters. *rSLDS* and *SLDS* match their synthetic 1 settings exactly, the `ssm` library (v0.0.1), $D{=}6$, three discrete states, 25 iterations, and a frozen identity emission.

Ablations. Each ablation changes exactly one component of ML-dLDS and leaves the rest at the shared settings above. We first test no penalty, which replaces the elastic-net $\boldsymbol{S}$-step with plain least squares. We then test ℓ_2 only, which removes the ℓ_1 term and keeps the ridge, so we solve $(\boldsymbol{A}^\top\boldsymbol{A} + \lambda\boldsymbol{I})^{-1}\boldsymbol{A}^\top\boldsymbol{b}$ with $\lambda{=}2.0$, a nonnegative clip, and row normalization. After that, fixed $\boldsymbol{c}$ disables the $\boldsymbol{c}$-step and holds $\boldsymbol{c}{=}1$. We also test single LDS, which sets $K{=}1$, $\Delta{=}1$, and single LDS with lags, which sets $K{=}1$, $\Delta{=}4$ and reuses one shared circuit across lags. On synthetic 2 the circuits already separate across lags, so we drop the orthogonality term of equation 4 and set $\lambda_2{=}0$. The $\boldsymbol{S}$ penalty then reduces to the lag-weighted ℓ_1 with the small ℓ_2 for stability, the elastic-net setting we use above.

Model selection. Sweeping the number of circuits K at the true maximum lag, and the maximum lag Δ at the true K, both the circuit correlation $\mathrm{corr}(\boldsymbol{F}, \hat{\boldsymbol{F}})$ and the lag support peak at the generative values (Fig. 10).

Undefined cells. Full VAR is closed-form and outputs no lag support, so its support cell is not applicable. The rSLDS and SLDS models produce no lag assignment, so their support is not applicable and their circuits come from the identity-emission observed dynamics. Single LDS and single LDS with lags use $K{=}1$, so support is undefined and their circuit correlation is the best single-circuit match to any true circuit.

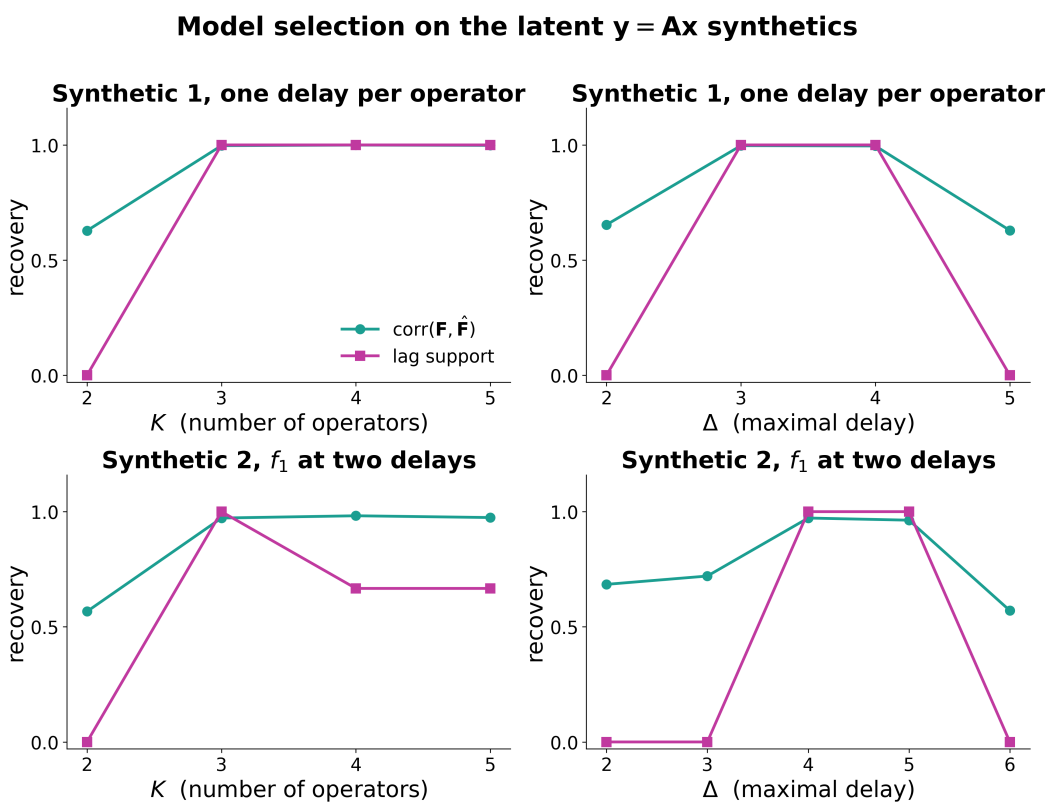

Figure 10: **Model selection on the latent $y = Ax$ synthetics.** Recovery against the number of operators K at the true maximal delay, and against the maximal delay Δ at the true K, one fit per cell. Teal is the operator correlation $\mathrm{corr}(\boldsymbol{F}, \hat{\boldsymbol{F}})$ after latent-basis alignment. Magenta is the lag support, the fraction of operators whose recovered delay set matches the truth. **(A,B)** Synthetic 1, one delay per operator. **(C,D)** Synthetic 2, operator $\boldsymbol{f}_1$ acting at two delays. Each cell is a single fit with latent dimension d at the true value, $P = 60$ observed dimensions, $T = 350$, and no observation noise. Both metrics peak at the true K and the true Δ.

G LAG-WEIGHT PROFILE ROBUSTNESS

We test whether the lag-weight profile w_δ of equation 4 drives delay selection, since w_δ sets how much a longer lag costs and thus which lag a circuit keeps when several fit equally. We sweep the profile shape through $w_\delta = \mathrm{base}^{\,\delta}$, where $\mathrm{base}{=}1$ gives the uniform profile that reduces to plain ℓ_1, moderate values give an increasing profile, and large values give a steep profile. We change only w_δ and hold λ_1, λ_c, λ_3, the schedule, and the seed fixed, and we keep $\lambda_2{=}0$ on both synthetic datasets. We run the sweep at seed 0 on synthetic 1 and synthetic 2, and we score lag support match, circuit correlation $\mathrm{corr}(\boldsymbol{f}_k, \hat{\boldsymbol{f}}_k)$, and reconstruction error $|\boldsymbol{x} - \hat{\boldsymbol{x}}|/|\boldsymbol{x}|$. On both datasets the recovery stays flat for every profile up to $\mathrm{base}{=}5$, where synthetic 1 holds support at 100% and synthetic 2 holds the two-delay support of $\boldsymbol{f}_1$ at $t{-}1$ and $t{-}3$. A steeper profile breaks the recovery, since $\mathrm{base}{\geq}10$ drops synthetic 1 support to 67% and collapses the two-delay circuit of synthetic 2 onto its short lag, and it raises reconstruction error about fourfold on both (Fig. 7).

Noise robustness. We test sensitivity to observation noise by adding i.i.d. Gaussian noise of standard deviation σ to the generated observations, then running the same detect-cluster-refine pipeline with fixed hyperparameters. Figure 12 reports circuit recovery ($\boldsymbol{F}$ correlation), lag support, and reconstruction relative error against σ, as mean and standard deviation over three seeds. Recovery stays near perfect for $\sigma \leq 0.1$, where $\boldsymbol{F}$ correlation exceeds 0.98 and the lag support matches the ground truth. Accuracy declines through $\sigma = 0.2$ and collapses by $\sigma = 0.3$, once noise overwhelms

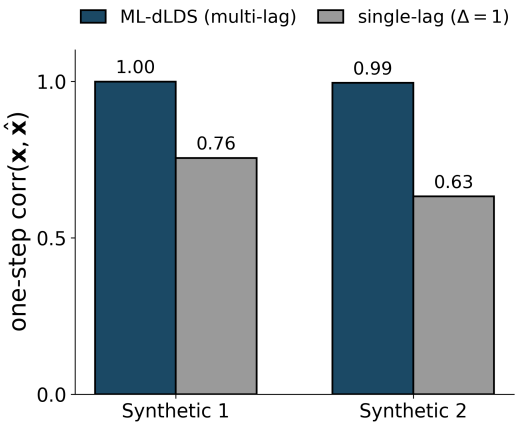

Figure 11: **Held-out one-step prediction, multi-lag versus single-lag.** On the held-out tail of each synthetic, the coefficients are re-solved from past states alone, then the next step is predicted. The multi-lag model predicts the held-out data better than a single-lag restriction that forces $\Delta = 1$.

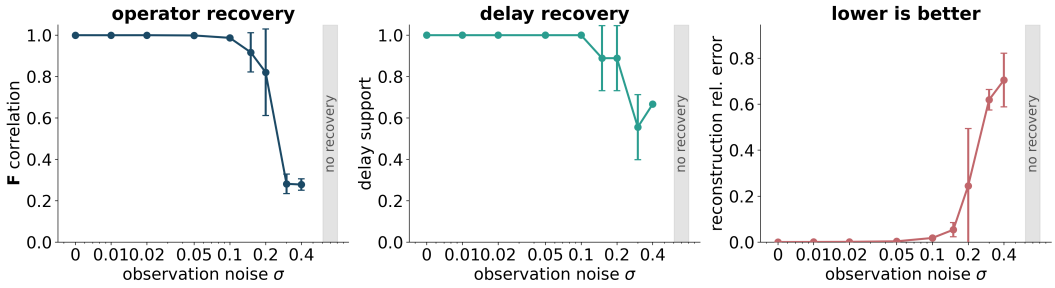

Figure 12: Noise robustness on synthetic 1. Circuit recovery ($\boldsymbol{F}$ correlation), lag support, and reconstruction relative error against observation noise σ, as mean and standard deviation over three seeds. Recovery stays near perfect for $\sigma \leq 0.1$ and collapses by $\sigma = 0.3$. The detector finds no clean solo window at $\sigma = 0.5$, shown as the no-recovery band.

the solo windows the initialization relies on. At $\sigma = 0.5$ the detector finds no clean solo window on any seed, so the method reports no fit, marked as the no-recovery band in Figure 12.

H Neural Data Experiment A, additional information

Data and preprocessing. We used one session of the Neuropixels dataset of Steinmetz et al. (2019), obtained through the DANDI Archive (dandiset 000017). The session contained $N = 326$ simultaneously recorded units across three areas, primary visual cortex (VISp), lateral posterior thalamus (LP), and secondary motor cortex (MOs). We estimated firing rates in 50 ms time bins and kept units with a mean rate above 0.5 Hz. We applied no z-scoring and no temporal smoothing to the signal used for the fit, and we treated the session as one continuous stream rather than concatenating isolated trials. Behavioral variables (choice, stimulus contrast, feedback) came from the trial table and were aligned to stimulus onset for the trial-locked analyses.

ML-dLDS configuration. We fit ML-dLDS with latent dimension $P = 17$, $K = 10$ circuits, and $\Delta = 6$ delays. The coefficient step used the spgl1 solver in its residual (BPDN) mode with $\sigma = 0.7$, no ℓ_2 penalty on $\boldsymbol{S}$, and an orthogonality penalty on the rows of $\boldsymbol{S}$ with weight 1.0. The orthogonality penalty was the key lever against single-circuit collapse; without it one circuit absorbed most of the coefficient mass. Larger latent dimensions ($P \geq 25$) reintroduced the collapse, so we kept $P = 17$. The fit used 9 of the 10 circuits and spread activation across delays $\delta = 1$ to 6.

Baselines. All latent baselines used the same latent dimension $P = 17$ and the same number of circuits or states $K = 10$, and every method was scored by reconstruction relative error in the observed space, $\|\boldsymbol{y} - \hat{\boldsymbol{y}}\|/\|\boldsymbol{y}\|$. The single-lag dLDS is our own pipeline restricted to one delay ($\Delta = 1$), so it isolates the contribution of multiple lags while matching every other setting. The full vector autoregression (VAR) fits a separate dense $N \times N$ circuit per lag in the observed space by ridge regression, and the single-lag VAR is its one-lag version. The switching baselines (rSLDS, SLDS) were fit with the `ssm` package using a reduced latent of dimension $P = 17$ and a learned orthogonal emission, with recurrent and standard transitions respectively.

Task decoding. We decoded task variables from the latent trajectories averaged in post-onset windows. We report cross-validated accuracy with stratified five-fold splits and compare against the majority-class chance level per variable. Time-resolved decoding used a sliding window aligned to stimulus onset. Per-area decoding restricted the reconstruction to the units of each area.

Table 3: Baseline comparison on the real Neuropixels session (dataset 000017). Reconstruction relative error in observed space, lower is better, the same metric across all methods. ML-dLDS matches the full VAR while using far fewer parameters and improves on the single-lag, switching, and region-based baselines.

model	recon. rel. err.
ML-dLDS (ours)	**0.35**
full VAR	0.35
rSLDS	0.60
SLDS	0.60
single-lag dLDS	0.61
single-lag VAR	0.66
MRM-GP (Li et al., 2024a)	0.68

I COVID-19 TEMPORAL SEGMENTATION, ADDITIONAL INFORMATION

We tested whether the recovered circuits occupy distinct stretches of time rather than co-activating. For each circuit k we summed the coefficient magnitude over delays, $m_k(t) = \sum_\delta |c_{k,\delta}(t)|$, and measured two quantities. The first is the mean share of the most active circuit per week, $\frac{1}{T}\sum_t \max_k m_k(t)/\sum_j m_j(t)$, which reached 0.997. The second is the mean pairwise correlation between the circuit activations m_k, which reached -0.05. We compared both against a null that circularly shifts each circuit's activation by an independent random offset, which preserves each circuit's own active pattern while it breaks the alignment between circuits. Across 5000 shifts the dominant-circuit share sat far above the null mean of 0.907 ($p < 0.001$), and the pairwise correlation sat below the null mean ($p = 0.001$). A single circuit governs each moment, so ML-dLDS segments the record into non-overlapping regimes with no regime labels supplied.

J RECOVERABILITY OF THE LAG FACTORIZATION

The dynamics in equation 2 enter through the product $c_{kt}^{(\delta)} s_k^{(\delta)}$, so the model sees the activation and the lag gain only through their product. Here we state what fixes this split and when the active lags are recoverable. The recovery is conditional, and we give the conditions under which the fit returns the intended factors.

Assumptions. (A1) *Scale fixing:* each row of $\boldsymbol{S}$ is normalized to unit maximum, absorbing the gain into $\boldsymbol{c}$. (A2) *Lag sparsity:* each circuit's true row $\boldsymbol{S}_{k,:}$ is supported on few lags. (A3) *Incoherence:* the past-state columns $\{\boldsymbol{x}_{t-\delta}\}_{\delta=1}^{\Delta}$ are mutually incoherent, with coherence below the standard sparse-recovery threshold (Donoho & Elad, 2003; Tropp, 2004). (A4) *Sharing:* $\boldsymbol{S}$ is shared across time, so the effective design pools all t.

Proposition (lag-support recovery). Under (A1)–(A4), the lag-sparsity solve recovers the true lag support of each $\boldsymbol{S}_{k,:}$, and the scale split $c_{kt}^{(\delta)}\, s_k^{(\delta)}$ is fixed to one representative per equivalence

Table 4: **Wall-clock of a single synthetic fit as the recording length T and the latent dimension P grow.** Each cell is one fit at $K = 3$ circuits and $\Delta = 3$ lags, on one node with two 24-core Intel Xeon Cascade Lake 6248R processors at 3.0 GHz, using a single core. The time grows roughly linearly in T and mildly in P, consistent with the per-timepoint SPGL1 coefficient solve that runs once per time bin at every outer pass.

T	P	wall-clock (s)
300	6	21.5
300	12	26.2
300	24	17.5
900	6	20.7
900	12	20.6
900	24	40.2
2700	6	51.4
2700	12	52.1
2700	24	80.8

class. When (A3) fails, that is, when two past states carry near-identical information, the support is not identifiable from the data alone, and the penalty returns the sparsest support consistent with the observations.

Scale ambiguity. For a circuit k and lag δ, the pair $(c_{kt}^{(\delta)}, s_k^{(\delta)})$ and the pair $(\alpha\, c_{kt}^{(\delta)}, s_k^{(\delta)}/\alpha)$ with $\alpha > 0$ give the same product and the same dynamics. We remove this freedom by normalizing each row of $\boldsymbol{S}$ to unit maximum and absorbing the scale into $\boldsymbol{c}$ (Alg. 1), which fixes one representative per class. After normalization $s_k^{(\delta)}$ reads as a relative lag gain and $c_{kt}^{(\delta)}$ carries the time course.

Lag support. With the scale fixed, the open question is which lags each circuit uses, the support of the row $\boldsymbol{S}_{k,:}$. At one time point the update in equation 2 is a weighted combination of the past states $\{\boldsymbol{x}_{t-\delta}\}_{\delta=1}^{\Delta}$, so recovering the support is a sparse-recovery problem over a dictionary built from these past states. When the past states are mutually incoherent and the true activation is sparse, the lag-sparsity penalty on $\boldsymbol{S}$ drives each circuit onto its few active lags, and the orthogonality penalty spreads circuits across different lags. We note that because $\boldsymbol{S}$ is shared across time, evidence at other moments fixes the lag even where one moment stays ambiguous.

Inherent ambiguity of the data. The past states are not always incoherent. When the data change little across the relevant lags, or when two past states carry nearly the same information, more than one lag assignment inherently reproduces the observations equally well. In this regime no assignment is more correct than another, since each explains the data. This ambiguity is a property of the data and not of the model or the solver. Faced with several equally valid solutions, ML-dLDS resolves the choice by the sparsity assumption, reporting the assignment on the fewest lags.

The second synthetic dataset reaches this regime. Circuit $\boldsymbol{f}_1$ acts at $t-1$ and $t-3$ with equal gain, so on a fraction of seeds both the two-lag solution and a one-lag solution fit the data, and the sparsity assumption reports the single lag (App. D.2). The collapse tracks the data rather than the fit, since across seeds the correlation between the two past states does not predict which seeds collapse, so the effect reflects the inherent ambiguity we resolve by design rather than a limitation of the solver.

K RUNTIME

All fits ran on CPU. The full 000017 fit, with $P = 326$ units, $T = 65{,}199$ time bins at 50 ms, latent dimension $d = 17$, $K = 10$ operators, and $\Delta = 6$ delays, took approximately 54 minutes on one node with two 24-core Intel Xeon Cascade Lake 6248R processors (3.0 GHz, Rockfish cluster). The dominant cost is the per-timepoint SPGL1 coefficient solve, which runs once per time bin at every outer pass. A synthetic fit ($P = 6$, $T = 900$, $K = 3$, $\Delta = 3$) completes in about 25 seconds on the same hardware, so the cost scales with the number of units and the recording length rather than with the model order.

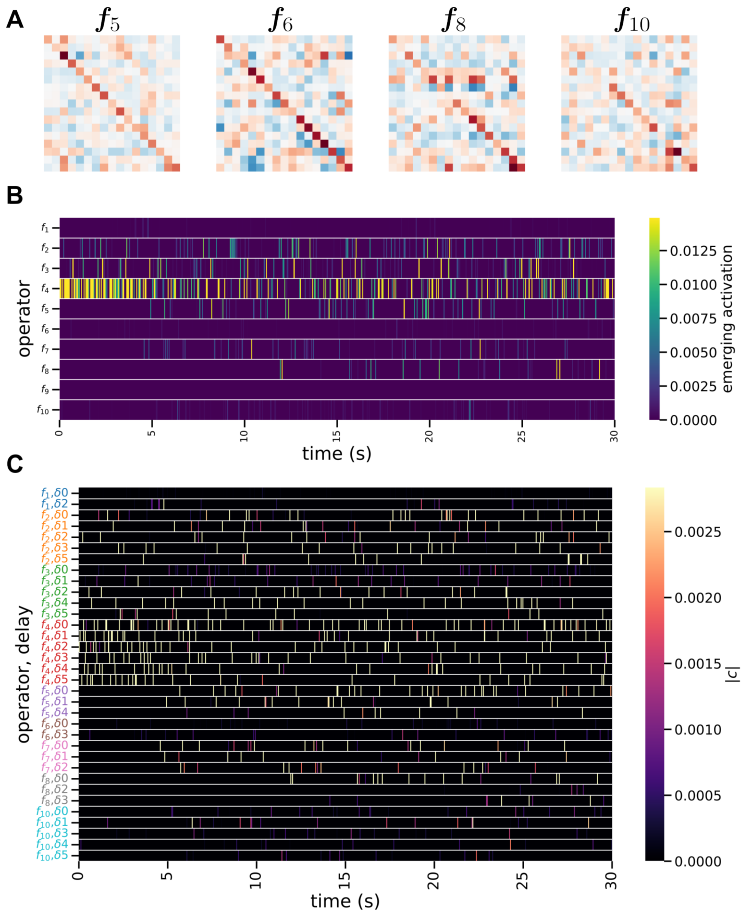

Figure 13: **Recovered latent circuits and their temporal activation on the mouse visual discrimination session. (A)** Four example shared circuits $\boldsymbol{f}_k$ as latent transition matrices; each circuit acts at every delay through its scaling profile $\boldsymbol{S}$, and the indicated δ marks its dominant lag. **(B)** Emerging dynamics, the total activation $\sum_\delta |c_{k,\delta}(t)|$ of each circuit over time, showing which circuits engage at each moment. **(C)** Active dynamics, the coefficient magnitude of each circuit and delay $(\boldsymbol{f}_k, \delta)$ over time, with time in seconds.

Computational complexity. We fit ML-dLDS by alternating minimization, and we state the cost of one outer pass in the model dimensions N (observed units), P (latent dimension), K (circuits), Δ (maximum lag), and T (time bins). The coefficient step infers the latent state $\boldsymbol{x}$ and the coefficients $\boldsymbol{c}$ together. At each time bin it builds a delayed dictionary of $K\Delta$ columns, each a product $s_k^{(\delta)} \boldsymbol{f}_k \boldsymbol{x}_{t-\delta}$ that costs $\mathcal{O}(P^2)$, then solves one SPGL1 problem over that dictionary whose iterations add matrix-vector products of size $P \times K\Delta$, so the step costs $\mathcal{O}(TK\Delta P^2)$. The $\boldsymbol{S}$-step updates the $K \times \Delta$ gains by projected gradient and reuses the same delayed products for its reconstruction gradient, so it also costs $\mathcal{O}(TK\Delta P^2)$. The $\boldsymbol{f}$-step solves one $P \times P$ ridge system per circuit, pooled over all time bins and delays, at $\mathcal{O}(KT\Delta P^2 + KP^3)$. The $\boldsymbol{A}$-step regresses the N observed channels on the P latent dimensions, at $\mathcal{O}(TNP + NP^2 + P^3)$. The dominant terms are $\mathcal{O}(TK\Delta P^2)$ from the coefficient and gain steps and $\mathcal{O}(TNP)$ from the emission step, both linear in T. In practice K and Δ stay small, so the recording length T and the per-bin SPGL1 solve set the runtime, which keeps the method usable at recording scale. The full 000017 fit, with $N = 326$, $P = 17$, $K = 10$, $\Delta = 6$, and $T \approx 65{,}000$ bins, ran in about 54 minutes on CPU.

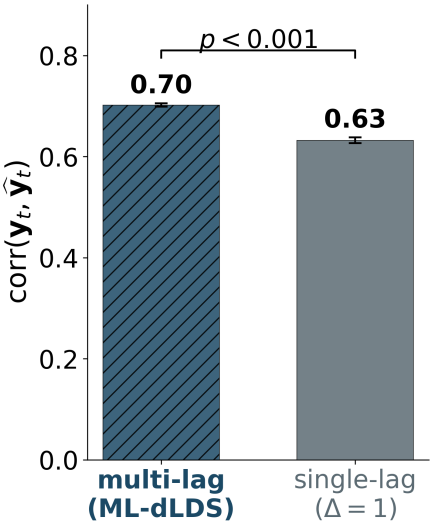

Figure 14: **One-step prediction improves with multiple delays.** Held-out one-step prediction correlation $\mathrm{corr}(\boldsymbol{y}_t, \widehat{\boldsymbol{y}}_t)$ for the full multi-lag model (ML-dLDS) and a single-lag restriction (Δ=1) of the same model. Bars show the mean and error bars the SEM across five folds. The multi-lag model reaches 0.70 ± 0.003 versus 0.63 ± 0.006 (paired t-test $p < 0.001$).