Inferring how internal brain state shapes neural responses with state-dependent diffusion models

Kaiwen Sheng[1,*], **Yiqi Jiang**[1,*], **Yuyang Song**[1,†], **Gaurav Tyagi**[1,†], **Haotian Ye**[1], **Fangzhao Zhang**[1]
Ruiming Du[2], **Pengxiao Wang**[3], **Stefano Ermon**[1], **Karl Deisseroth**[1,‡], **Anish Mitra**[1,‡]
[1]Stanford University [2]Cornell University [3]Peking University
{kwsheng, yqjiang, deissero, anishm}@stanford.edu

Abstract

Neural responses to the same stimulus can vary dramatically with fluctuations in the brain's internal state and the organism's behavioral state. Yet studying how brain state shapes neural responses is difficult: limited experimental recordings sparsely sample this joint space, leaving many biologically valid state–stimulus pairings unobserved. Existing methods either capture state-dependent modulation with restricted model assumptions or synthesize neural activity without explicitly controlling internal brain state. We introduce **BrainStateDiff**, a conditional diffusion framework that models the spatiotemporal distribution of future neural activity and behavior given external stimuli and recent brain states. Motivated by two gaps in finite recordings, missing state–stimulus combinations and limited repeats of observed conditions, BrainStateDiff introduces two state-aware sampling strategies: *cross-state sampling*, which recombines observed states and stimuli to synthesize plausible neural responses from unobserved pairings, and *matched-state sampling*, which resamples from observed pairings to capture local trial-to-trial variability. We evaluate BrainStateDiff on our newly collected wide-field calcium imaging data in mouse V1, electrophysiology data in monkey V4, and two-photon calcium imaging data in mouse striatum. Our results show that BrainStateDiff generates biologically meaningful samples that preserve the state–dependent structure of real recordings and expand neural response space coverage beyond the original recordings. BrainStateDiff supports both randomized passive designs, where cross-state sampling expands finite coverage over admissible state–stimulus pairs, and closed-loop designs, where neural state, behavior, and external conditions co-evolve through autoregressive generation and task design.[1]

1 Introduction

A central finding in systems neuroscience is that the same stimulus can evoke markedly different neural responses across trials [1–3] (Fig. 1A). This trial-to-trial variability is not random noise, but shaped by the brain's internal state, a moment-to-moment context of arousal, behavioral state (e.g., locomotion), and attention, at the time of the stimulus [4–8], making neural computation fundamentally state–dependent. In other words, neural activity is not a deterministic function of the external stimulus alone, but conditions jointly on the stimulus and the brain's internal state.

This state dependence creates a fundamental coverage problem for neural data analysis. Ideally, a dataset would sample enough of the joint space of external stimuli and internal states to characterize how neural responses vary across both dimensions. In practice, recordings are costly, time-consuming,

*Co-first authors. †Co-second authors. ‡Co-corresponding authors.

[1]Code is available at https://github.com/holmosaint/BrainStateDiff, and the mouse wide-field calcium imaging dataset is available at https://dandiarchive.org/dandiset/002040.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

and constrained by experimental design, so they cover only a sparse subset of this space (Fig. 1B). We view this sparsity as creating two distinct gaps. First, many biologically plausible combinations of neural state and external stimuli may be rare or absent in a finite recording [6–12], yielding a *combinatorial coverage gap*. Second, even an observed state–stimulus pair can produce a distribution of responses due to intrinsic stochasticity, incomplete state measurements, or observation noise [13–15], yielding a *local variability gap*. These missing combinations limit the ability to analyze how external stimuli are represented across the diverse internal states *in vivo*.

A generative model that learns the conditional distribution of neural activity given external stimuli and internal state can turn a finite recording into a controllable simulator of the underlying neural system. Rather than being restricted to state–stimulus combinations that happened to be observed experimentally, such a model can synthesize responses in under-sampled regions of the joint space of external stimuli and internal brain states, enabling analyses beyond the empirical recorded data. Yet, prior neural generative models synthesize spiking activity unconditionally or conditioned on task, stimulus, or behavioral covariates [16, 17], without explicit conditioning on internal brain states.

To facilitate the study of how brain states affect neural responses, we introduce **BrainStateDiff** (Fig. 1C), a conditional diffusion framework [18–20] that infers state-dependent neural responses given a state–stimulus pairing, including combinations unseen in the experimental recordings. Given a recent neural state ($\mathbf{s}$) and external stimuli ($\mathbf{u}$), BrainStateDiff models the distribution of subsequent neural activity and behavior ($\mathbf{y}$). This turns sparse recordings into a controllable generator of the conditional response distribution. BrainStateDiff introduces two complementary sampling strategies that target the two gaps introduced above. *Cross-state sampling* recombines observed neural states with observed external stimuli to generate responses for state–stimulus pairings that might be absent from the original dataset, directly addressing the combinatorial coverage gap (Fig. 1D). Such recombinations are valid in most randomized designs when stimulus identity is chosen independently of the animal's instantaneous neural state. Cross-state sampling therefore expands effective coverage of the conditional response manifold without requiring new experiments. *Matched-state sampling*, in contrast, resamples responses around observed state–stimulus pairs to enrich local variability, tackling the local variability gap (Fig. 1D). In closed-loop applications, generated neural activity and behavior update the subsequent state autoregressively, while task dynamics update the external condition.

Our main contributions are: (1) We introduce an evaluation protocol combining matched-state and cross-state sampling methods to test state–stimulus compositional generalization. (2) We develop BrainStateDiff as a high-dimensional stochastic response model for this problem and show its stronger state-dependent generalization than baseline models in both passive and closed-loop settings. (3) We evaluate BrainStateDiff across three recording modalities and two species, including a newly collected mouse wide-field calcium imaging dataset. (4) We show that BrainStateDiff generates biologically plausible state-dependent responses that preserve stimulus-, state-, and behavior-related structure in real recordings, validated through visual classification, natural-image reconstruction, and behavioral forecasting. (5) We show that state-aware sampling realistically expands neural response-space coverage beyond the original recordings.

2 Related Work

Internal-state modulation of neural responses. Neuroscience studies have shown that neural responses are modulated by internal brain and behavioral states. Earlier studies demonstrated that locomotion, arousal, and pupil-related variables modulate sensory response [5–8]. More recent studies combine large-scale neural recordings with behavioral tracking, and use encoding analyses to relate neural variability to behavioral and brain-wide state fluctuations [2, 4, 21, 22]. However, existing approaches mainly analyze experimentally observed state–response covariation or model response distributions with restrictive parametric likelihoods. This limits their ability to flexibly sample biologically plausible responses under rare or unobserved state–stimulus pairings.

Generative latent variable models in neuroscience. Generative latent variable models have been used to characterize neural activity, including variational autoencoders (VAE) [22–24], sequential VAEs [25–29], flow-based models [30], state-space models [31–35], and generative adversarial networks [17]. More recently, diffusion models have been applied to neural data synthesis and representation learning, including latent diffusion models for spiking activity [16] and diffusion-based analyses of behavior-relevant neural dynamics in widefield calcium recordings [36]. However,

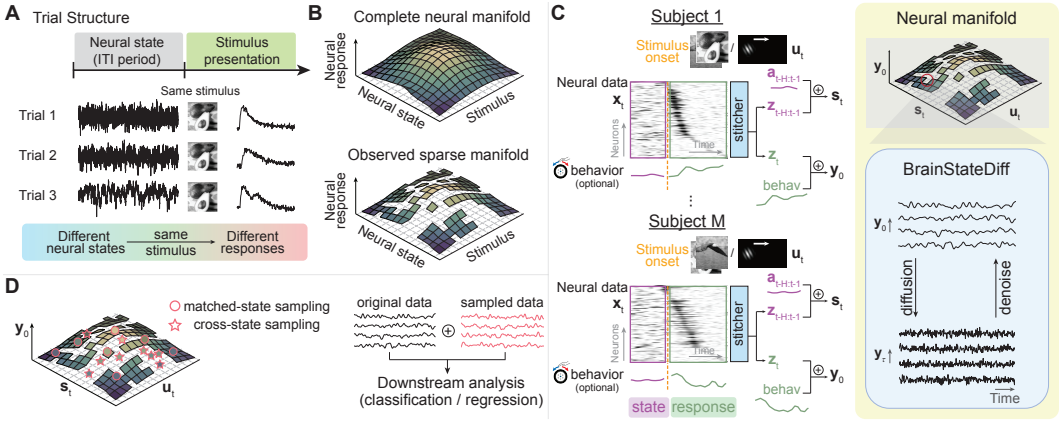

Figure 1: **Overview of BrainStateDiff, a conditional diffusion framework for state-dependent neural response generation. A.** The same external stimuli evoke different responses under different neural states. **B.** Real recordings cover the state–stimulus space sparsely, leaving many neural state–stimulus pairings unobserved. **C.** BrainStateDiff first stitches recordings from different sessions and subjects into a shared representation space, then trains a diffusion model conditioned on neural state, external stimuli, and recording identity. **D.** At inference, *cross-state sampling* recombines observed states and inputs to cover rare or absent pairings, while *matched-state sampling* reuses observed pairs to sample local variability. Synthetic and real data are combined for downstream analysis.

the state variables in these models are generally inferred from trial or drawn from a learned prior distribution, leaving no direct support of holding a brain state fixed or re-pairing it with a different stimulus. So these models are not directly designed to address how the same stimulus would evoke different responses under different pre-stimulus states. Furthermore, existing neural diffusion models generate responses under fixed conditions, but are not designed to explore unobserved state–input pairings or simulate feedback-driven dynamics. BrainStateDiff enables both controllable cross-state sampling and closed-loop rollout, extending neural diffusion from fixed-state sampling to controllable generation and closed-loop rollout.

3 Methods

3.1 Framework overview

We formulate synthetic neural data generation as a conditional generative process (Fig. 1C). Given the current neural state, external stimuli, and recording identity (such as subject and session IDs), BrainStateDiff models the distribution of future neural activity and, when available, future behavior. Let $\mathbf{s}_t$ denote the neural state at time t, let $\mathbf{u}_t$ denote the external stimuli (such as an image presentation or auditory cue) to the subject, and let m denote subject or session ID. We seek to learn

$$p_\theta(\mathbf{y}_{\text{future}} \mid \mathbf{s}_t, \mathbf{u}_t, m), \tag{1}$$

where θ collects the learnable parameters of the generative model, and $\mathbf{y}_{\text{future}}$ denotes the future outputs to be simulated. Depending on the dataset, $\mathbf{y}_{\text{future}}$ may consist of future neural activity alone or a joint future target that includes both neural activity and behavior.

Conditioning jointly on $\mathbf{s}_t$ and $\mathbf{u}_t$ addresses the combinatorial coverage gap by allowing valid state–stimulus pairings beyond those recorded, while modeling p_θ as a distribution addresses the local variability gap by sampling multiple plausible responses for the same pairing.

3.2 Neural state representation

We represent the neural state as a function of recent neural activity and/or behavior (when available) over a fixed history window of length H (Fig. 1C). Because the neural observation space typically differs across subjects or sessions, each observation is first mapped onto a shared representation space through a subject/session-specific stitcher E_m (Fig. 1C), which is instantiated either as a learnable

projection or as the encoder of a pretrained VAE

$$\mathbf{z}_t = E_m(\mathbf{x}_t), \tag{2}$$

where $\mathbf{x}_t$ is the raw neural activity at time t and $\mathbf{z}_t$ denotes the latent representation on the shared neural latent manifold across sessions and subjects.

The state variable is the concatenation of recent neural history and optional auxiliary signals

$$\mathbf{s}_t = [\mathbf{z}_{t-H:t-1}, \mathbf{a}_{t-H:t-1}], \tag{3}$$

where $\mathbf{a}_{t-H:t-1}$ collects any additional history variables, such as history behaviors, available in a given dataset. The key is that $\mathbf{s}_t$ captures the recent dynamical context that modulates the neural response to the external stimuli.

3.3 BrainStateDiff: brain-state-dependent conditional diffusion model

We model the conditional distribution of future outputs with a diffusion model whose denoiser is conditioned on the current state, external stimuli, and recording identity. Let $\mathbf{y}_0$ denote a clean future target. Following the standard denoising diffusion probabilistic model formulation [19], during training, we sample a diffusion timestep τ and perturb the target by the forward noising process

Table 1: **Symbol notation.**

Symbol	Definition
$\mathbf{x}_t$	Recorded neural activity at time t
$\mathbf{z}_t$	Latent embedding of $\mathbf{x}_t$
$\mathbf{s}_t$	Pre-target history conditioning generation
$\mathbf{y}_t^+$	Clean future neural activity and behavior
$\mathbf{y}_{t,\tau}^+$	Diffusion-corrupted $\mathbf{y}_t^+$ at step τ

$$\mathbf{y}_\tau = \sqrt{\bar{\alpha}_\tau}\,\mathbf{y}_0 + \sqrt{1-\bar{\alpha}_\tau}\,\boldsymbol{\epsilon}, \qquad \boldsymbol{\epsilon} \sim \mathcal{N}(0, I), \tag{4}$$

where $\bar{\alpha}_\tau$ is the cumulative noise-schedule coefficient that controls how much clean signal remains at diffusion timestep τ. The denoiser receives the noisy target together with conditioning variables and predicts the clean sample directly

$$\hat{\mathbf{y}}_0 = f_\theta(\mathbf{y}_\tau, \mathbf{s}_t, \mathbf{u}_t, m, \tau). \tag{5}$$

Using x_0-prediction [37], the training objective is

$$\mathcal{L}_{\text{diff}} = \mathbb{E}_{(\mathbf{y}_0, \mathbf{s}_t, \mathbf{u}_t, m) \sim p_{\text{data}},\, \tau,\, \boldsymbol{\epsilon}} \left[\| f_\theta(\mathbf{y}_\tau, \mathbf{s}_t, \mathbf{u}_t, m, \tau) - \mathbf{y}_0 \|_2^2 \right]. \tag{6}$$

BrainStateDiff can simulate long-horizon dynamics by generating short future segments sequentially. Given an initial state, the model samples the next response segment of length κ from $p_\theta(\mathbf{y}_{t:t+\kappa-1} \mid \mathbf{s}_t, \mathbf{u}_t)$. The generated segment is then used to update the state, producing a new condition $\mathbf{s}_{t+\kappa}$ for the next sampling step. This autoregressive mode allows the model to simulate extended neural activity and, if available, behaviors.

3.4 Cross-state and matched-state synthetic sampling

At inference time, synthetic data generation depends not only on the learned diffusion model, but also on how state–stimulus pairs are selected (Fig. 1D). To fill the combinatorial coverage gap, we introduce *cross-state* sampling. Let t_0 denote the start of generation and $\mathbf{u}^{\text{ext}}$ denote the initially assigned external input, such as an image in passive viewing or a trial cue in a closed-loop task. For a fixed recording identity m, we independently select an initial history and an external input:

$$p_{\text{cross}}^{\text{init}}(\mathbf{s}_{t_0}, \mathbf{u}^{\text{ext}} \mid m) = p_{\text{pool}}(\mathbf{s}_{t_0} \mid m)\, p_{\text{data}}(\mathbf{u}^{\text{ext}} \mid m). \tag{7}$$

Eq. 7 defines a cross-state initialization proposal for both passive and closed-loop experiments in which the experimental design assigns the initial image or cue independently of ongoing neural activity. Cross-state sampling thereby expands coverage of valid state–stimulus pairings that are rare or absent in the limited real recording. When the experimental design restricts allowable pairings, an experiment-specific admissibility distribution $q_{\text{adm}}(\mathbf{s}_{t_0}, \mathbf{u}^{\text{ext}} \mid m)$ can be used instead.

During the closed-loop experiments, neural, behavioral, and cursor histories are retained together, preserving their observed relationships. We distinguish the assigned input $\mathbf{u}^{\text{ext}}$ from the full current

task condition $\mathbf{u}_t$, which also includes behavior-dependent variables. The synthetic responses are thus generated as

$$\tilde{\mathbf{y}}_t^+ \sim p_\theta(\mathbf{y}_t^+ \mid \mathbf{s}_t, \mathbf{u}_t, m). \tag{8}$$

For closed-loop generation, $\tilde{\mathbf{y}}_t^+$ contains a jointly sampled segment of future neural latents and behavior of length κ. The conditions for the next segment are updated through

$$(\mathbf{s}_{t+\kappa}, \mathbf{u}_{t+\kappa}) = \mathcal{T}_m\left(\mathbf{s}_t, \mathbf{u}_t, \tilde{\mathbf{y}}_t^+\right), \tag{9}$$

where $\mathcal{T}_m$ combines the known task dynamics with the rolling-history update. For instance, in the striatal task, generated wheel velocity updates cursor position through the cursor dynamics based on the experimental design. The generated neural latents, wheel velocity, and updated cursor trajectory are then appended to the conditioning history, replacing the oldest frames, while the trial cue remains fixed (Appendix D.4). Thus, subsequent states and task conditions evolve jointly through generated behavior and the task dynamics, rather than being resampled from Eq. 7.

As a complementary regime, *matched-state* sampling initializes generation using state–stimulus pairs observed in the original data:

$$p^{\text{init}}_{\text{match}}(\mathbf{s}_{t_0}, \mathbf{u}^{\text{ext}} \mid m) = p_{\text{data}}(\mathbf{s}_{t_0}, \mathbf{u}^{\text{ext}} \mid m). \tag{10}$$

Both sampling regimes use the same conditional generator and, for closed-loop experiments, the same update rule. Matched-state sampling tackles the local variability gap by generating additional plausible responses for observed state–stimulus pairs, capturing variability from intrinsic stochasticity.

4 Results

Across the following experiments, we first use a simulated synthetic dataset with ground-truth conditional response distributions to show that BrainStateDiff can recover missing state–stimulus distributions. We then evaluate BrainStateDiff across three recording modalities and two species, spanning repeated stimuli, single-presentation stimuli, and long-horizon dynamics, including one newly collected dataset. For each real-data setting, we demonstrate that BrainStateDiff not only generates realistic neural activity, but also faithfully expands the neural response space.

4.1 BrainStateDiff recovers missing state–stimulus response distributions

We validated BrainStateDiff with simulated responses from a smooth, heteroscedastic Gaussian distribution conditioned on a continuous state $s \in [0, 1]$ and stimulus $u \in [0, 1]$, to mirror the real neural manifold that nearby states and stimuli induce similar response distributions [13, 15, 38]

$$y \mid s, u \sim \mathcal{N}\left(\mu(s, u), \sigma^2(s, u)\right). \tag{11}$$

We introduced a combinatorial coverage gap by removing a diagonal band of (s, u) pairings from the training data, while preserving broad marginal coverage of both variables (Fig. 2A; Appendix A).

We then trained and sampled from BrainStateDiff. The two sampling strategies probe different parts of this space. Matched-state sampling drew conditioning pairs from the empirical joint distribution $p_{\text{data}}(s, u)$, keeping generation on observed coordinates. In this region, BrainStateDiff accurately inferred the conditional response distributions around empirical state–stimulus pairs. Cross-state sampling drew s and u independently from their empirical marginals, producing conditioning pairs inside the missing band. In the missing region, cross-state sampling recovered the smooth mean surface $\mu(s, u)$ and the heteroscedastic standard deviation surface $\sigma(s, u)$ (Fig. 2B–C). We compare the generated samples in both observed and missing regions from BrainStateDiff with that from a state-dependent generalized linear model (GLM; Appendix A.4). Although the GLM baseline can accurately infer the mean in observed regions, its estimation errors increase substantially in the missing area (Fig. 2D). In addition, compared with BrainStateDiff, the GLM exhibits larger errors in estimating the standard deviation across both observed and missing regions (Fig. 2E). Together, these results demonstrate that BrainStateDiff can learn local response variability where observations exist and, through cross-state sampling, combine observed states and stimuli to generate plausible responses at valid but unobserved pairings.

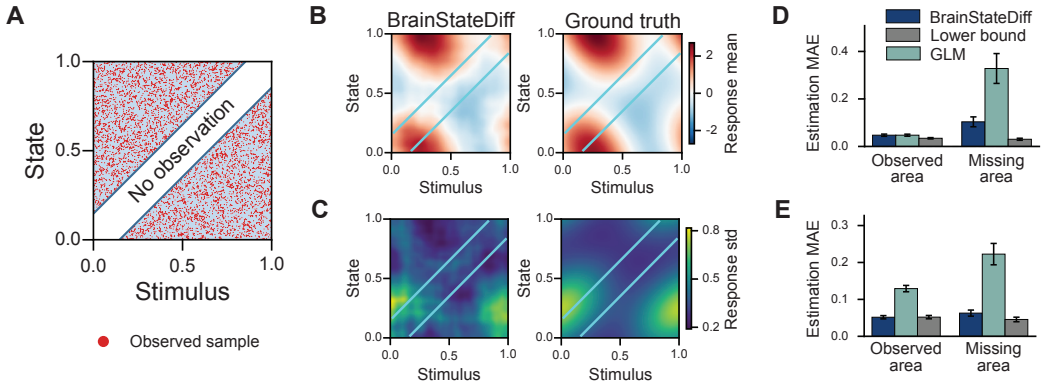

Figure 2: **BrainStateDiff recovers observed and missing state–stimulus response distribution on synthetic dataset with cross-state and matched-state sampling. A.** The synthetic task setup. Blue indicates the observed region and the outlined band indicates the missing region. Red dots: observed samples. **B.** BrainStateDiff-generated mean surface and true mean surface. The region between the cyan lines is not observed. **C.** BrainStateDiff-generated standard deviation surface and true standard deviation surface. **D.** Conditional mean estimation error in observed and missing cells. **E.** Conditional standard-deviation estimation error in observed and missing cells. Error bars: s.e.m.

4.2 BrainStateDiff generates realistic stimulus-specific spatiotemporal calcium responses beyond recorded state–stimulus pairings

To evaluate whether BrainStateDiff realistically expands neural response coverage beyond repeated stimulus presentations, we collected wide-field calcium imaging from V1 of two mice viewing natural images from THINGS [39, 40]. The dataset contains five sessions per mouse, with each image presented for 750 ms and repeated 80 times per animal (Fig. 3A; Appendix B). BrainStateDiff is conditioned on a neural state $\mathbf{s}_t$ extracted from pre-stimulus activity using a pretrained multi-subject Stitch-VAE, an image stimulus representation $\mathbf{u}$ from pretrained ViT-MAE tokens [41], and a learned subject embedding for the recording identity m (Appendix B.2).

We first examined whether BrainStateDiff samples capture the response variability (Fig. 3A) and stimulus-specific structure observed in the real recordings. BrainStateDiff samples drawn under cross-state sampling reproduced this trial-to-trial heterogeneity without collapsing onto the trial mean (Fig. 3B). Both matched-state and cross-state sampling recovered the image-evoked trial averaged spatial pattern of V1 activation (Fig. 3C). To quantify stimulus-specific fidelity, we z-scored each pixel-time bin across trials and measured how strongly each synthetic trial-averaged response correlated with the real trial average for the same image versus the other 77 images (Fig. 3D). Both matched-state and cross-state samples showed substantially higher correlations with the real response to the same image while keeping the correlations with responses to other images near zero (Fig. 3D), indicating that the model is not producing a generic V1 activation pattern shared across stimuli.

We then evaluated whether synthetic responses preserve stimulus information at the single-trial level without simply reproducing the training responses. We trained a 78-way image classifier on synthetic responses alone and evaluated it on held-out real trials (Fig. 3E; Appendix B.5). For cross-state sampling, we enumerated all state–stimulus pairings to avoid subsampling bias and to evaluate generation under recombined pairings not directly observed in the recordings. Both matched-state and cross-state samples achieved above-chance decoding on real data ($\sim$1.3% chance), indicating that the synthetic responses contain stimulus-discriminative structure aligned with real neural activity. Across BrainStateDiff training epochs, matched-state accuracy increased monotonically, whereas cross-state accuracy rose quickly, plateaued, and then declined. This pattern is consistent with generalization-to-memorization transition described for diffusion models training, where an intermediate training window can generate novel high-quality samples before later training increases memorization of the empirical training distribution [42]. Because matched-state sampling uses observed state–stimulus pairings, it can benefit from memorization. Cross-state sampling instead requires generalization to recombined pairings that were not observed together, so excessive memorization can reduce its performance. We therefore selected the checkpoint with the best validation performance for all subsequent analyses to avoid the memorization issue in diffusion model training.

Having established response-level fidelity, we next tested whether BrainStateDiff samples provide complementary coverage of the empirical state–stimulus distribution. We used single-trial image decoding as a coverage assay: realistic synthetic responses should improve decoding on held-out real data when combined with real training data. Synthetic-only training remained slightly below the real-only baseline (Fig. 3F; Appendix B.4), consistent with prior observations that generative samples alone are less sample-efficient than real data [43–45]. After real-data finetuning, however, both matched-state and cross-state samples improved over the real-only baseline (Fig. 3F), indicating that the generated responses contain complementary structure not captured by the finite recordings. Cross-state sampling yielded larger and more consistent gains, suggesting that recombining observed images with broader neural states fills state–stimulus coverage gaps beyond local resampling of observed pairings. BrainStateDiff has the highest reported mean synthetic-only accuracy at all three factors under both sampling strategies. Its finetuned mean is highest at all three factors under matched-state sampling and at $40\times$ and $78\times$ under cross-state sampling, whereas LFADS is slightly higher at cross-state $1\times$ (Table S2). Under both sampling strategies, BrainStateDiff yielded response-space coverage gains closer to those from adding held-out real responses than LFADS (Table S11).

To test if BrainStateDiff uses brain state in a biologically meaningful way, for each held-out trial, we use the same BrainStateDiff model to generate two responses to the same stimulus: one conditioned on the trial's true pre-stimulus state, one on a state from a different trial. Both BrainStateDiff and LFADS benefit from the true state on held-out trials. BrainStateDiff shows a consistently larger

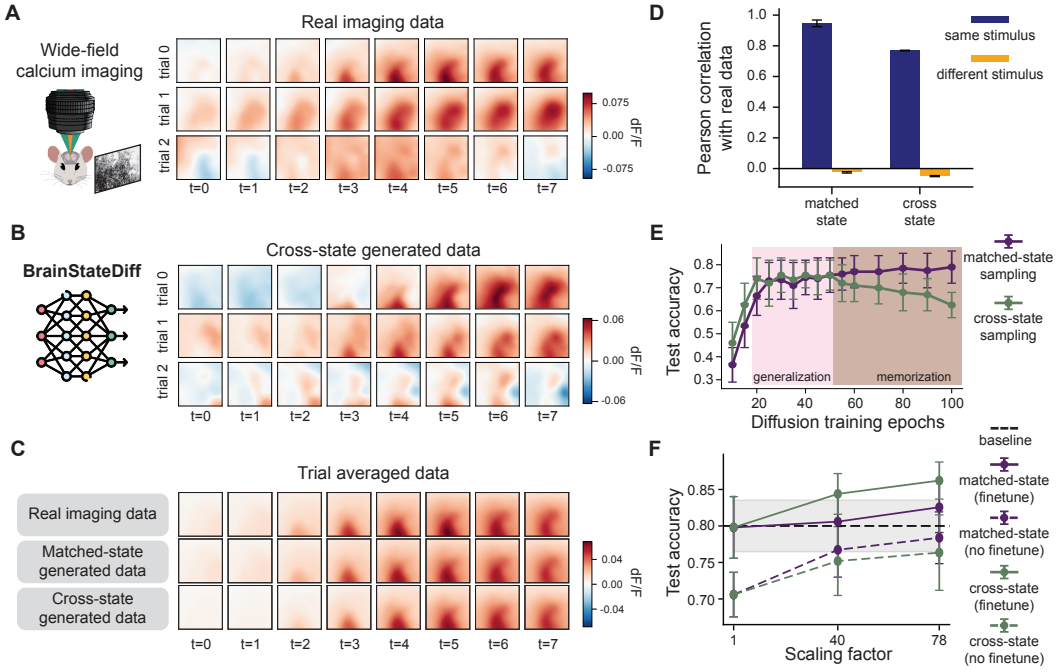

Figure 3: **BrainStateDiff generates realistic state-dependent responses and expands response coverage in mouse V1. A.** Experimental setup (left) and three example real trials from one mouse in response to the same natural image (right); $t = 0$ denotes stimulus onset and each time step is $\sim$24 ms. **B.** Three example single-trial responses to the same natural image generated by BrainStateDiff under cross-state sampling. **C.** Trial-averaged responses to the same natural image of real data (top), BrainStateDiff with matched-state sampling (middle), and cross-state sampling (bottom). **D.** Spatiotemporal Pearson correlation between real and synthetic trial-averaged responses for same-image and different-image pairings and for matched-state and cross-state sampling; mean $\pm$ s.e.m. across mice. **E.** Test accuracy of a 78-way classifier trained on synthetic data only, as a function of the BrainStateDiff training epoch from which the synthetic trials were sampled, for matched-state and cross-state sampling; mean $\pm$ s.e.m., $n = 2$ mice. **F.** Test accuracy as a function of the synthetic-to-real training set size ratio (scaling factor). Dashed lines show synthetic-only training; solid lines show synthetic pretraining followed by real-data finetuning. Mean $\pm$ s.e.m., $n = 2$ mice.

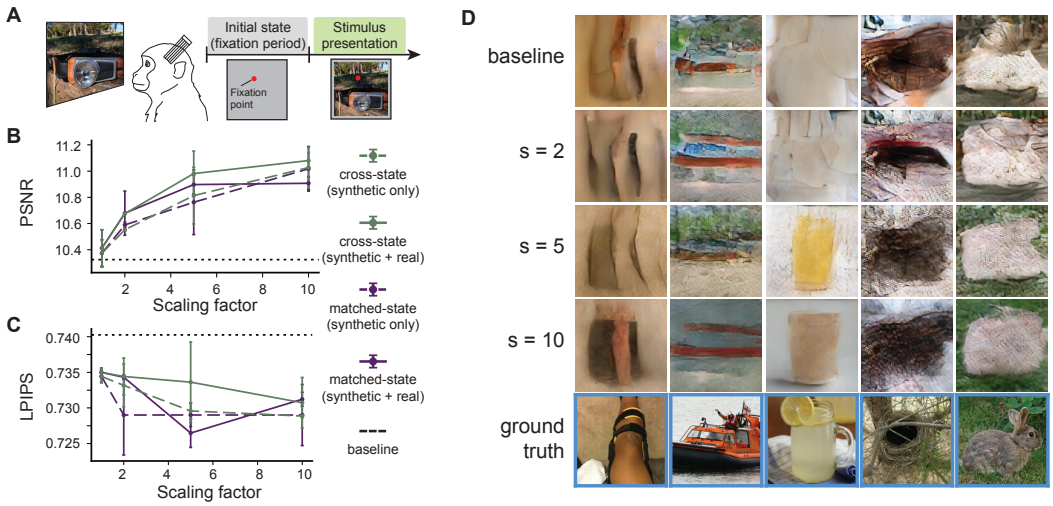

Figure 4: **BrainStateDiff generates realistic state-dependent responses and expands response coverage in monkey V4. A.** Experimental setup of monkeys passively viewing natural images while multi-unit activity was recorded from V4. **B.** Image-reconstruction performance measured by PSNR (higher is better). **C.** Image-reconstruction performance measured by LPIPS (lower is better). **D.** Representative held-out image reconstructions from V4 responses. Rows show the baseline, BrainStateDiff-augmented reconstructions at various synthetic-to-real scaling factors, and the corresponding ground-truth images.

numerical improvement across energy score, RMSE, and response correlation, indicating that its generated responses depend strongly on trial-specific brain state (Table S6).

The large performance gap on the mouse widefield V1 dataset should not be interpreted purely as the advantage of diffusion over other generative mechanisms. The V1 widefield calcium responses are dense $H \times W \times T$ movies whose two-dimensional organizations, including spatial proximity and retinotopic structure, carries stimulus information. BrainStateDiff preserves this organization through its spatially structured latent representation and spatiotemporal denoising architecture, whereas other baselines operate on flattened $(H \times W) \times T$ representations and therefore lack an explicit inductive bias for local two-dimensional structure (see Appendix B.6 for details). Spatially adapting these baseline families could improve their performance on the widefield V1 dataset, but is beyond the scope of the present work. Our newly collected V1 dataset exposes this important challenge for future spatiotemporal neural generative models.

Together, the above results show that BrainStateDiff generates realistic, stimulus-specific calcium responses even for state–stimulus recombinations not directly observed in the experiment. Because repeated presentations already provide relatively broad response coverage for each image, the stronger gains from cross-state sampling support BrainStateDiff's role as a state-aware generator that expands the remaining combinatorial state–stimulus coverage of the training distribution.

4.3 BrainStateDiff expands image-dependent V4 electrophysiology response coverage

We then evaluated BrainStateDiff in the single-presentation stimulus setting using multi-unit electrophysiology from monkey V4 in the THINGS ventral stream spiking dataset (TVSD) [46]. In this experiment, two monkeys passively viewed natural images while V4 multi-unit activity was recorded (Fig. 4A; Appendix C). Unlike the wide-field calcium imaging dataset, TVSD contains broad natural-image coverage, but each training image is shown only once and associated with a single measured pre-stimulus state and a single evoked V4 response. This design limits the empirical response coverage in two ways: local trial-to-trial variability around each image is unobserved, and most biologically valid state–image pairings are absent from the recordings.

We first verified that BrainStateDiff preserved image-dependent V4 response structure. When the neural activity matrix was sorted by the maximally responsive neuron for each stimulus, BrainStateDiff samples preserved the stimulus-specific structure observed in real V4 responses (Fig. S1).

This suggested that BrainStateDiff is not merely producing generic V4 activity, but instead captures image-dependent response structure in the recorded population.

To test whether BrainStateDiff samples realistically fill both the local variability gap around each image and the combinatorial gap across state–image pairings, we evaluated natural-image reconstruction from V4 activity on held-out data. We trained the state-of-the-art latent-diffusion reconstructor from the STSBench [47] on synthetic-only or synthetic plus real data, and compared with the real-only baseline from the benchmark. The reconstructor is trained only on TVSD images and conditions solely on V4 activity, with no externally pretrained image model. The same reconstructor is used for the real-only baseline and every augmented condition, so differences in reconstruction quality reflect the training set rather than an image prior. Reconstruction quality was evaluated using the two metrics reported in the STSBench: peak signal-to-noise ratio (PSNR; higher is better), which measures pixel-level fidelity, and learned perceptual image patch similarity (LPIPS; lower is better), which measures perceptual similarity in the feature space of an image-recognition network.

Both matched-state and cross-state sampling alone improved reconstruction from held-out real V4 responses relative to the baseline (Fig. 4B–C; Table S9), suggesting that BrainStateDiff samples realistically expand image-dependent V4 response coverage by filling both local variability around each image and unobserved state–image pairings. Cross-state sampling achieved larger gains in PSNR (Fig. 4B), whereas matched-state sampling improved LPIPS more (Fig. 4C), suggesting that the two sampling strategies provide complementary coverage of the response distribution. Combining synthetic and real data further improved PSNR relative to synthetic-only training, while LPIPS showed less additional benefit, suggesting that synthetic responses provide useful coverage but do not fully replace real measurements. Consistent with these trends, larger synthetic-to-real scaling factors produced visually cleaner reconstructions with smoother backgrounds and sharper edges (Fig. 4D). BrainStateDiff outperforms all five baselines under matched-state sampling and the two state-dependent baselines shown under cross-state sampling across all reported scaling factors (Table S3). On held-out trials, true-state conditioning benefits both BrainStateDiff and LFADS, with consistently larger gains for BrainStateDiff across energy score, RMSE, and response correlation, indicating strong dependence on trial-specific brain state (Table S7). Direct log-determinant and participation-ratio gains show that BrainStateDiff cross-state augmentation expands both covariance volume and effective dimensionality beyond matched-state augmentation, with gains more comparable to adding held-out real trials than LFADS (Table S12).

4.4 BrainStateDiff generates plausible neural–behavioral trajectories beyond recorded state–stimulus pairings

Finally, we evaluated BrainStateDiff on two-photon calcium imaging from mouse dorsolateral striatum during a wheel-turning task [33] (Fig. 5A; Appendix D). Unlike the passive viewing datasets, the external condition here evolves with the animal's behavior: wheel velocity changes the cursor position, which in turn defines the next task state. This closed-loop structure tests whether BrainStateDiff can model state-dependent neural and behavioral dynamics over variable-length trials.

Because neural activity, behavior, and stimulus co-evolve in this setting, generating neural activity conditioned on a fixed behavioral trace would not capture the closed-loop task structure. We therefore implemented BrainStateDiff as an autoregressive latent diffusion model (Appendix D.3). Starting from an observed pre-cursor-onset history, BrainStateDiff used recent neural latents and wheel velocity as the state $\mathbf{s}$, and the initial cursor position and cue identity as the task condition $\mathbf{u}$. It then jointly sampled future neural latents and wheel velocity, updated cursor position through the known task dynamics, and fed the updated neural–behavioral history back into the model (Appendix D.4).

To assess whether generated rollouts (Fig. 5B) lie within the neural manifold observed in real trials, we fit a UMAP [48] embedding on real activity from one representative session and projected synthetic samples into the same space. Synthetic samples occupied the manifold smoothly, preserving behavioral organization while increasing trajectory density (Fig. 5C). To test whether the synthetic neural-behavioral trajectories preserve the state-dependent neural and behavioral dynamics, we tested behavior forecasting performance on held-out real trials. We trained linear decoders to predict wheel velocity at forecasting horizons $K = 0, 3, 7$, with $K = 0$ corresponding to instantaneous behavioral decoding. Synthetic-only training surpassed the real-only baseline once enough generated trajectories were used, and synthetic pretraining followed by real-data finetuning further improved performance across horizons (Fig. 5D). These results indicated that BrainStateDiff rollouts preserve

neural structure predictive of real behavior while adding coverage beyond the finite recordings. The gains increased with forecasting horizon, with the largest improvements at longer horizons (Fig. 5E), suggesting that the generated trajectories capture temporal neural–behavioral dynamics rather than only instantaneous correlations. BrainStateDiff achieved the highest mean synthetic-only R^2 across all forecasting horizons and scaling factors under both matched-state and cross-state sampling (Table S4). Across held-out trials, BrainStateDiff and LFADS both benefit from the true initial state. BrainStateDiff's larger gains in energy score, RMSE, and response correlation support its use of trial-specific state (Table S8). Under both sampling strategies, BrainStateDiff had response-space coverage gains closer to those from adding held-out real responses than LFADS (Table S13).

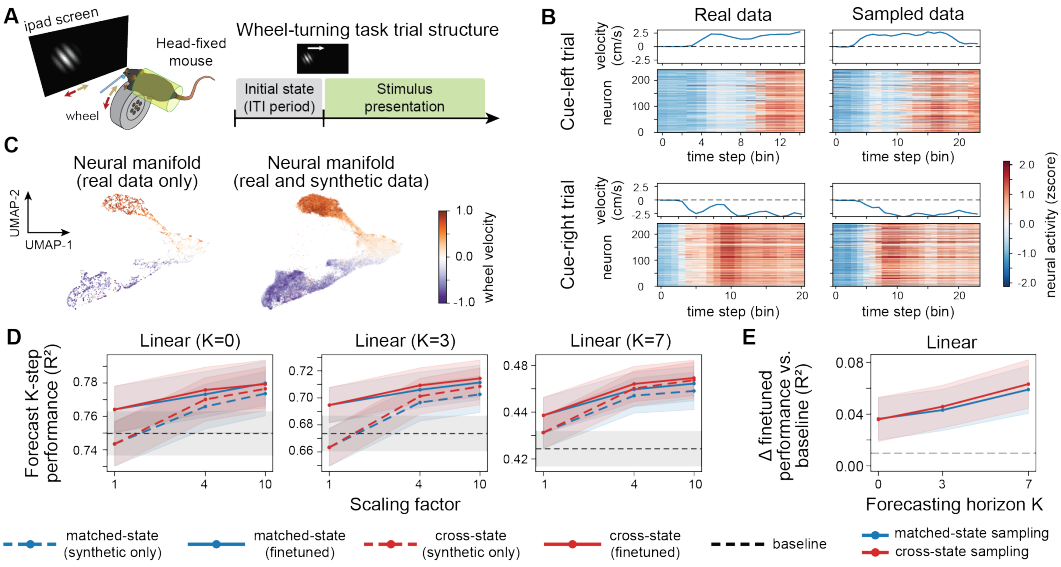

Figure 5: **BrainStateDiff generates realistic state-dependent responses and expands response coverage in mouse striatum. A.** Experimental setup of head-fixed mice turning a wheel to move a cursor to the center of the screen for a reward. **B.** Example real and BrainStateDiff-generated single trials for left-cue and right-cue trials. For each trial, the top trace shows wheel velocity and the heatmap shows z-scored neural activity. **C.** UMAP visualization of neural activity from one sample session. The embedding is fit on real data and then applied to synthetic data; dots are colored by wheel velocity. **D.** Linear forecasting performance (R^2) at forecasting horizons $K = 0, 3, 7$, plotted as a function of synthetic dataset size. Black dashed line: real-only baseline. Colored dashed lines: synthetic-only training. Colored solid lines: synthetic pretraining followed by real-data finetuning. Blue: matched-state sampling. Red: cross-state sampling. **E.** Finetuned linear decoding gains over the real-only baseline as a function of forecasting horizon K. Mean $\pm$ s.e.m., $n = 13$ sessions.

5 Discussion

Across three recording modalities and two species, BrainStateDiff was able to model realistic brain-state-dependent neural response distributions. BrainStateDiff introduces two complementary sampling strategies: cross-state sampling, which recombines observed states and inputs to address missing combinatorial coverage, and matched-state sampling, which enriches local variability around observed state–stimulus pairings. Both sampling strategies produce realistic yet diverse samples.

Several limitations and future directions remain. Unrestricted recombination should not be used when experimental assignment does not support the proposed state–input pairings. Adaptive or closed-loop settings instead require explicit transition dynamics as in our striatum experiment, while the validity of initial history–cue recombination remains dependent on assignment, temporal ordering, and overlap. Second, future versions of BrainStateDiff could incorporate richer state variables, such as arousal, pupil dynamics, attention, motivation, and task engagement, rather than relying only on recent neural and behavioral history. Finally, future work can support the evaluation of whether BrainStateDiff improves a broader range of analyses, including neural representation learning, uncertainty estimation, and hypothesis testing about state-dependent coding.

Acknowledgments and Disclosure of Funding

This research was supported in part by the National Institute of Mental Health (NIMH award K08MH131888), the Burroughs Wellcome Fund, and The Whitehall Foundation. We thank Ethan Trepka and Yuekun Dai for helpful discussions and feedback on this work. The authors declare no competing interests or conflicts of interest.

References

[1] Amos Arieli, Alexander Sterkin, Amiram Grinvald, and Ad Aertsen. Dynamics of ongoing activity: explanation of the large variability in evoked cortical responses. *Science*, 273(5283): 1868–1871, 1996. doi: 10.1126/science.273.5283.1868.

[2] Carsen Stringer, Marius Pachitariu, Nicholas Steinmetz, Charu Bai Reddy, Matteo Carandini, and Kenneth D. Harris. Spontaneous behaviors drive multidimensional, brainwide activity. *Science*, 364(6437):eaav7893, 2019. doi: 10.1126/science.aav7893.

[3] Pierre O. Boucher, Tian Wang, Laura Carceroni, Gary Kane, Krishna V. Shenoy, and Chandramouli Chandrasekaran. Initial conditions combine with sensory evidence to induce decision-related dynamics in premotor cortex. *Nature Communications*, 14(1):6510, 2023. doi: 10.1038/s41467-023-41752-2.

[4] Simon Musall, Matthew T. Kaufman, Ashley L. Juavinett, Steven Gluf, and Anne K. Churchland. Single-trial neural dynamics are dominated by richly varied movements. *Nature Neuroscience*, 22:1677–1686, 2019. doi: 10.1038/s41593-019-0502-4.

[5] Matthew J. McGinley, Martin Vinck, Jacob Reimer, Renata Batista-Brito, Edward Zagha, Cathryn R. Cadwell, Andreas S. Tolias, Jessica A. Cardin, and David A. McCormick. Waking state: Rapid variations modulate neural and behavioral responses. *Neuron*, 87(6):1143–1161, 2015. doi: 10.1016/j.neuron.2015.09.012.

[6] Cristopher M. Niell and Michael P. Stryker. Modulation of visual responses by behavioral state in mouse visual cortex. *Neuron*, 65(4):472–479, 2010. doi: 10.1016/j.neuron.2010.01.033.

[7] Jacob Reimer, Emmanouil Froudarakis, Cathryn R. Cadwell, Dimitri Yatsenko, George H. Denfield, and Andreas S. Tolias. Pupil fluctuations track fast switching of cortical states during quiet wakefulness. *Neuron*, 84(2):355–362, 2014. doi: 10.1016/j.neuron.2014.09.033.

[8] Martin Vinck, Renata Batista-Brito, Ulf Knoblich, and Jessica A. Cardin. Arousal and locomotion make distinct contributions to cortical activity patterns and visual encoding. *Neuron*, 86(3): 740–754, 2015. doi: 10.1016/j.neuron.2015.03.028.

[9] Artur Luczak, Peter Barthó, and Kenneth D. Harris. Spontaneous events outline the realm of possible sensory responses in neocortical populations. *Neuron*, 62(3):413–425, 2009. doi: 10.1016/j.neuron.2009.03.014.

[10] Lloyd E. Russell, Mehmet Fişek, Zidan Yang, Lynn Pei Tan, Adam M. Packer, Henry W. P. Dalgleish, Selmaan N. Chettih, Christopher D. Harvey, and Michael Häusser. The influence of cortical activity on perception depends on behavioral state and sensory context. *Nature Communications*, 15(1):2456, 2024. doi: 10.1038/s41467-024-46484-5.

[11] Alexander S. Ecker, Philipp Berens, R. James Cotton, Manivannan Subramaniyan, George H. Denfield, Cathryn R. Cadwell, Stelios M. Smirnakis, Matthias Bethge, and Andreas S. Tolias. State dependence of noise correlations in macaque primary visual cortex. *Neuron*, 82(1): 235–248, 2014. doi: 10.1016/j.neuron.2014.02.006.

[12] Shailaja Akella, Peter Ledochowitsch, Joshua H. Siegle, Hannah Belski, Daniel D. Denman, Michael A. Buice, Severine Durand, Christof Koch, Shawn R. Olsen, and Xiaoxuan Jia. Deciphering neuronal variability across states reveals dynamic sensory encoding. *Nature Communications*, 16(1):1768, 2025. doi: 10.1038/s41467-025-56733-w.

[13] Shreya Saxena and John P. Cunningham. Towards the neural population doctrine. *Current Opinion in Neurobiology*, 55:103–111, 2019. doi: 10.1016/j.conb.2019.02.002.

[14] Juan A. Gallego, Matthew G. Perich, Lee E. Miller, and Sara A. Solla. Neural manifolds for the control of movement. *Neuron*, 94(5):978–984, 2017. doi: 10.1016/j.neuron.2017.05.025.

[15] Matthew G. Perich, Devika Narain, and Juan A. Gallego. A neural manifold view of the brain. *Nature Neuroscience*, 28(8):1582–1597, 2025. doi: 10.1038/s41593-025-02031-z.

[16] Jaivardhan Kapoor, Auguste Schulz, Julius Vetter, Felix Pei, Richard Gao, and Jakob H. Macke. Latent diffusion for neural spiking data. In *Advances in Neural Information Processing Systems 37 (NeurIPS)*, 2024.

[17] Jiangrong Shen, Kejun Wang, Wei Gao, Jian K. Liu, Qi Xu, Gang Pan, Xiaodong Chen, and Huajin Tang. Temporal spiking generative adversarial networks for heading direction decoding. *Neural Networks*, 184:106975, 2025. doi: 10.1016/j.neunet.2024.106975.

[18] Prafulla Dhariwal and Alexander Nichol. Diffusion models beat GANs on image synthesis. In *Advances in Neural Information Processing Systems 34 (NeurIPS)*, 2021.

[19] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems 33 (NeurIPS)*, 2020.

[20] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[21] Atika Syeda, Lin Zhong, Renee Tung, Will Long, Marius Pachitariu, and Carsen Stringer. Facemap: a framework for modeling neural activity based on orofacial tracking. *Nature Neuroscience*, 27(1):187–195, 2024. doi: 10.1038/s41593-023-01490-6.

[22] Ryan V Raut, Zachary P Rosenthal, Xiaodan Wang, Hanyang Miao, Zhanqi Zhang, Jin-Moo Lee, Marcus E Raichle, Adam Q Bauer, Steven L Brunton, Bingni W Brunton, and J Nathan Kutz. Arousal as a universal embedding for spatiotemporal brain dynamics. *Nature*, 647(8089): 454–461, 2025. doi: 10.1038/s41586-025-09544-4.

[23] Ding Zhou and Xue-Xin Wei. Learning identifiable and interpretable latent models of high-dimensional neural activity using pi-VAE. In *Advances in Neural Information Processing Systems 33 (NeurIPS)*, 2020.

[24] Jonnathan Singh Alvarado, Jack Goffinet, Valerie Michael, William Liberti III, Jordan Hatfield, Timothy Gardner, John Pearson, and Richard Mooney. Neural dynamics underlying birdsong practice and performance. *Nature*, 599(7886):635–639, 2021. doi: 10.1038/s41586-021-04004-1.

[25] 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, Jaimie M. Henderson, Krishna V. Shenoy, L. F. Abbott, and David Sussillo. Inferring single-trial neural population dynamics using sequential auto-encoders. *Nature Methods*, 15(10):805–815, 2018. doi: 10.1038/s41592-018-0109-9.

[26] Mohammad Reza Keshtkaran, Andrew R. Sedler, Raeed H. Chowdhury, Raghav Tandon, Diya Basrai, Sarah L. Nguyen, Hansem Sohn, Mehrdad Jazayeri, Lee E. Miller, and Chethan Pandarinath. A large-scale neural network training framework for generalized estimation of single-trial population dynamics. *Nature Methods*, 19:1572–1577, 2022. doi: 10.1038/s41592-022-01675-0.

[27] Cole Hurwitz, Akash Srivastava, Kai Xu, Justin Jude, Matthew G. Perich, Lee E. Miller, and Matthias H. Hennig. Targeted neural dynamical modeling. In *Advances in Neural Information Processing Systems 34 (NeurIPS)*, 2021.

[28] Feng Zhu, Harrison A Grier, Raghav Tandon, Changjia Cai, Anjali Agarwal, Andrea Giovannucci, Matthew T Kaufman, and Chethan Pandarinath. A deep learning framework for inference of single-trial neural population dynamics from calcium imaging with subframe temporal resolution. *Nature Neuroscience*, 25(12):1724–1734, 2022. doi: 10.1038/s41593-022-01189-0.

[29] Emily L. Sylwestrak, YoungJu Jo, Sam Vesuna, Xiao Wang, Blake Holcomb, Rebecca H. Tien, Doo Kyung Kim, Lief Fenno, Charu Ramakrishnan, William E. Allen, Ritchie Chen, Krishna V. Shenoy, David Sussillo, and Karl Deisseroth. Cell-type-specific population dynamics of diverse reward computations. *Cell*, 185(19):3568–3587, 2022. doi: 10.1016/j.cell.2022.08.019.

[30] Mohammad Bashiri, Edgar Y. Walker, Konstantin-Klemens Lurz, Akshay Kumar Jagadish, Taliah Muhammad, Zhiwei Ding, Zhuokun Ding, Andreas S. Tolias, and Fabian H. Sinz. A flow-based latent state generative model of neural population responses to natural images. In *Advances in Neural Information Processing Systems 34 (NeurIPS)*, 2021.

[31] Amber Hu, David Zoltowski, Aditya Nair, David Anderson, Lea Duncker, and Scott Linderman. Modeling latent neural dynamics with Gaussian process switching linear dynamical systems. In *Advances in Neural Information Processing Systems 37 (NeurIPS)*, 2024.

[32] Omid G Sani, Bijan Pesaran, and Maryam M Shanechi. Dissociative and prioritized modeling of behaviorally relevant neural dynamics using recurrent neural networks. *Nature Neuroscience*, 27(10):2033–2045, 2024. doi: 10.1038/s41593-024-01731-2.

[33] Yiqi Jiang, Kaiwen Sheng, Yujia Gao, E. Kelly Buchanan, Yu Shikano, Seung Je Woo, Yixiu Zhao, Tony Hyun Kim, Fatih Dinc, Scott Linderman, and Mark Schnitzer. Extracting task-relevant preserved dynamics from contrastive aligned neural recordings. In *Advances in Neural Information Processing Systems 38 (NeurIPS)*, 2025.

[34] Aditya Nair, Amit Vinograd, Mengyu Liu, George Mountoufaris, Scott Linderman, and David J Anderson. The neural computation of affective internal states in the hypothalamus: A dynamical systems perspective. *Neuron*, 113(23):3887–3907, 2025. doi: 10.1016/j.neuron.2025.11.003.

[35] Victor Geadah, Amin Nejatbakhsh, David Lipshutz, Jonathan Pillow, and Alex Williams. Modeling neural activity with conditionally linear dynamical systems. In *Advances in Neural Information Processing Systems 38 (NeurIPS)*, 2025.

[36] Yule Wang, Chengrui Li, Weihan Li, and Anqi Wu. Exploring behavior-relevant and disentangled neural dynamics with generative diffusion models. In *Advances in Neural Information Processing Systems 37 (NeurIPS)*, 2024.

[37] Alexander Quinn Nichol and Prafulla Dhariwal. Improved denoising diffusion probabilistic models. In *International conference on machine learning*, pages 8162–8171. PMLR, 2021.

[38] Nikolaus Kriegeskorte and Rogier A. Kievit. Representational geometry: integrating cognition, computation, and the brain. *Trends in Cognitive Sciences*, 17(8):401–412, 2013. doi: 10.1016/j.tics.2013.06.007.

[39] Martin N. Hebart, Adam H. Dickter, Alexis Kidder, Wan Y. Kwok, Anna Corriveau, Caitlin Van Wicklin, and Chris I. Baker. THINGS: A database of 1,854 object concepts and more than 26,000 naturalistic object images. *PLOS ONE*, 14(10):e0223792, 2019. doi: 10.1371/journal.pone.0223792.

[40] Laura M. Stoinski, Jonas Perkuhn, and Martin N. Hebart. THINGSplus: New norms and metadata for the THINGS database of 1854 object concepts and 26,107 natural object images. *Behavior Research Methods*, 56(3):1583–1603, 2024. doi: 10.3758/s13428-023-02110-8.

[41] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[42] Tony Bonnaire, Raphaël Urfin, Giulio Biroli, and Marc Mézard. Why diffusion models don't memorize: The role of implicit dynamical regularization in training. In *Advances in Neural Information Processing Systems 38 (NeurIPS)*, 2025.

[43] Shekoofeh Azizi, Simon Kornblith, Chitwan Saharia, Mohammad Norouzi, and David J. Fleet. Synthetic data from diffusion models improves ImageNet classification. *Transactions on Machine Learning Research*, 2023.

[44] Ruifei He, Shuyang Sun, Xin Yu, Chuhui Xue, Wenqing Zhang, Philip Torr, Song Bai, and Xiaojuan Qi. Is synthetic data from generative models ready for image recognition? In *International Conference on Learning Representations (ICLR)*, 2023.

[45] Mert Bülent Sarıyıldız, Karteek Alahari, Diane Larlus, and Yannis Kalantidis. Fake it till you make it: Learning transferable representations from synthetic ImageNet clones. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023.

[46] Paolo Papale, Feng Wang, Matthew W. Self, and Pieter R. Roelfsema. An extensive dataset of spiking activity to reveal the syntax of the ventral stream. *Neuron*, 113(4):539–553.e5, 2025. doi: 10.1016/j.neuron.2024.12.003.

[47] Ethan B. Trepka, Ruobing Xia, Shude Zhu, Sharif Saleki, Danielle Abreu Lopes, Stephen J. Niño Cital, Konstantin F. Willeke, Mindy Kim, and Tirin Moore. STSBench: A large-scale dataset for modeling neuronal activity in the dorsal stream of primate visual cortex. In *Advances in Neural Information Processing Systems 38 (NeurIPS), Datasets and Benchmarks Track*, 2025.

[48] Leland McInnes, John Healy, Nathaniel Saul, and Lukas Großberger. UMAP: Uniform manifold approximation and projection. *Journal of Open Source Software*, 3(29):861, 2018. doi: 10.21105/joss.00861.

[49] Taesung Park, Ming-Yu Liu, Ting-Chun Wang, and Jun-Yan Zhu. Semantic image synthesis with spatially-adaptive normalization. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (CVPR)*, pages 2337–2346, 2019.

[50] Diederik P Kingma and Max Welling. Auto-encoding variational Bayes. In *International Conference on Learning Representations (ICLR)*, 2014.

[51] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations (ICLR)*, 2019.

[52] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *International Conference on Learning Representations (ICLR)*, 2021.

[53] Albert Gu, Karan Goel, and Christopher Ré. Efficiently modeling long sequences with structured state spaces. In *International Conference on Learning Representations (ICLR)*, 2022.

[54] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations (ICLR)*, 2015.

[55] Yu Takagi and Shinji Nishimoto. High-resolution image reconstruction with latent diffusion models from human brain activity. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023.

[56] Richard Zhang, Phillip Isola, Alexei A. Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[57] Samy Bengio, Oriol Vinyals, Navdeep Jaitly, and Noam Shazeer. Scheduled sampling for sequence prediction with recurrent neural networks. In *Advances in Neural Information Processing Systems 28 (NeurIPS)*, 2015.

[58] Alex Lamb, Anirudh Goyal, Ying Zhang, Saizheng Zhang, Aaron Courville, and Yoshua Bengio. Professor forcing: A new algorithm for training recurrent networks. In *Advances in Neural Information Processing Systems 29 (NeurIPS)*, 2016.

[59] Manel Baradad, Jonas Wulff, Tongzhou Wang, Phillip Isola, and Antonio Torralba. Learning to see by looking at noise. In *Advances in Neural Information Processing Systems 34 (NeurIPS)*, pages 2556–2569, 2021.

[60] Luca Mazzucato, Alfredo Fontanini, and Giancarlo La Camera. Stimuli reduce the dimensionality of cortical activity. *Frontiers in Systems Neuroscience*, 10:11, 2016. doi: 10.3389/fnsys.2016.00011.

Appendix

A Synthetic Gaussian benchmark

A.1 Ground-truth response distribution

We constructed a controlled synthetic dataset with scalar state $s \in [0, 1]$, scalar stimulus $u \in [0, 1]$, and scalar response $y \mid s, u \sim \mathcal{N}(\mu(s, u), \sigma^2(s, u))$ as introduced in the main text, with smooth mean

$$\mu(s, u) = \sin(2\pi u) + 0.8\cos(2\pi s) + 0.6\sin(2\pi(s + u)) + 1.2su, \tag{S1}$$

and smooth positive standard deviation

$$\sigma(s, u) = 0.08 + \log(1 + \exp[-1 + 0.4\sin(2\pi s) + 0.3\cos(2\pi u) + 0.25\sin(2\pi(s - u))]) . \tag{S2}$$

This choice creates a response distribution that changes continuously with both state and stimulus, including both state-dependent shifts in the mean and heteroscedastic trial variability.

A.2 Observed and held-out regions

Training observations were sampled continuously from the state–stimulus plane while excluding a diagonal missing region

$$M = \{(s, u) : |s - u| \leq 0.15\}. \tag{S3}$$

The state and stimulus marginals therefore span the full interval $[0, 1]$, but the empirical joint distribution lacks the valid pairings inside M. We generated $5{,}000$ training observations and $1{,}000$ validation observations outside M. For analysis and visualization, the state–stimulus plane was also partitioned into a 10×10 grid. A grid cell was labeled missing when its center lay inside M and this yielded 72 observed cells and 28 missing cells.

A.3 Diffusion model

The denoiser was a four-hidden-layer MLP with width 128 and SiLU nonlinearities. Its inputs were the noisy scalar response y_τ, raw s and u, sinusoidal Fourier features of s and u using 8 frequencies $(1, \ldots, 2^7)$, and a 64-dimensional sinusoidal diffusion-timestep embedding. We trained an x_0-prediction diffusion model with a cosine noise schedule over $1{,}000$ training timesteps and sampled with deterministic DDIM using 50 inference steps. The response target y was standardized using the training-set mean and standard deviation before training and transformed back to the original response scale after sampling.

Optimization used AdamW with batch size 128, weight decay 10^{-4}, gradient clipping at 1.0, and a cosine learning-rate decay from 10^{-3} to 10^{-5} over $30{,}000$ optimization steps. We maintained an exponential moving average of model parameters with decay 0.999 and used the EMA weights for validation and sampling. Every 500 steps, we evaluated the EMA model on a held-out validation set using the denoising MSE and on a checkpoint-selection subset of missing cells using Gaussian KL divergence between generated and ground-truth conditional response distributions. This selection subset comprised half of the missing grid cells (14 of 28 cells), chosen by a fixed random split, with 200 generated samples per selected cell. The final checkpoint was the EMA model with the lowest average selection KL. For the reported grid evaluation, we generated 200 DDIM samples in every 10×10 state–stimulus cell and computed per-cell mean, standard deviation, and Gaussian KL metrics.

A.4 State-dependent GLM baseline

For the simulated dataset, we implemented a state-dependent generalized linear model that predicts the scalar response from smooth basis functions of the state and stimulus. Both s and u were expanded with 16 raised-cosine basis functions over $[0, 1]$, denoted $\phi_a(s)$ and $\varphi_c(u)$. The GLM included additive state and stimulus terms as well as all $\phi_a(s)\varphi_c(u)$ tensor-product interactions:

$$\eta(s, u) = \beta + \sum_a w_a^{(s)} \phi_a(s) + \sum_c w_c^{(u)} \varphi_c(u) + \sum_{a,c} W_{a,c} \phi_a(s)\varphi_c(u). \tag{S4}$$

The conditional mean was parameterized with a softplus link,

$$m(s,u) = b + \alpha \,\mathrm{softplus}(\eta(s,u)), \qquad y \mid s,u \sim \mathcal{N}\big(m(s,u), \sigma^2_{\mathrm{GLM}}\big)\,, \tag{S5}$$

where σ^2_{GLM} is a single homoscedastic variance parameterized through a learned log-variance. The response target was standardized using the training-set mean and standard deviation before fitting.

We optimized the Gaussian negative log likelihood with Adam, learning rate 10^{-3}, batch size 128, and gradient clipping at 1.0 for 15,000 steps, without additional regularization. To initialize the model near zero mean after standardization, the output offset b was initialized to $-\log 2$, so that $b + \mathrm{softplus}(0) = 0$. Every 500 steps, we evaluated validation negative log likelihood and the same checkpoint-selection KL used for the diffusion model: average Gaussian KL on a fixed half of the missing grid cells (14 of 28 cells), with 200 generated samples per selected cell. The final checkpoint was the model with the lowest selection KL. For the reported grid evaluation, we sampled from the learned Gaussian predictive distribution at 200 uniformly drawn (s,u) locations in each 10×10 analysis cell.

A.5 Matched-state and cross-state sampling

We drew matched-state samples by sampling conditioned on each observed training pair (s_i, u_i) 20 times, and we drew 20,000 cross-state samples by independently resampling states and stimuli from their empirical marginals. To estimate generated mean and standard-deviation surfaces, we also performed a grid sweep with 200 generated samples per 10×10 analysis cell, and a fine 50×50 grid sweep with 200 samples per fine cell for the heatmaps in Fig. 2B–C.

A.6 Fidelity metrics

For each analysis cell, we estimated the generated conditional mean $\hat{\mu}$ and standard deviation $\hat{\sigma}$ from generated samples and compared them to the ground-truth values μ and σ at the cell center. We report absolute mean error $|\hat{\mu} - \mu|$ and absolute standard-deviation error $|\hat{\sigma} - \sigma|$. Metrics were summarized separately over observed cells and missing cells; error bars in Fig. 2D–E denote s.e.m. across cells.

B Wide-field calcium imaging dataset of mouse visual cortex

B.1 Data acquisition

We performed wide-field calcium imaging in the right visual cortex of head-fixed Cux2-Cre GCaMP8m transgenic mice using a custom-built dual-wavelength system. Visual stimuli consisted of 78 natural images randomly selected from the THINGS dataset [39, 40]. Each image was presented to the animal's left eye for 750 ms with a randomized inter-trial interval of 1.5–2.5 s. Each image was repeated 80 times per animal across 5 recording days.

To separate calcium-dependent signals from hemodynamic and optical confounds, we alternated excitation with a 470 nm LED to capture calcium-dependent fluorescence and a 405 nm LED to capture an isosbestic reference channel. Raw movies were first motion-corrected using a cross-correlation-based alignment procedure. We then applied linear detrending to each pixel time series to remove slow photobleaching-related drift. Finally, we regressed out the isosbestic channel using a pixelwise linear model: for each pixel p, we fit $F_{470}(p,t) \approx a_p F_{405}(p,t) + b_p$ and used the residual as the corrected fluorescence signal.

We computed trial-wise $\Delta F/F$ separately for each stimulus presentation. For each trial, the baseline fluorescence F_0 was defined as the average of the first 20 frames of the trial, and we computed $\Delta F/F = (F - F_0)/F_0$ for subsequent analyses.

Real trials were partitioned into 80% training, 10% validation, and 10% test splits. The same partition was used across all components of the pipeline (stitch VAE, BrainStateDiff, and downstream decoder).

B.2 Stitch VAE

For the wide-field calcium imaging dataset, our diffusion model operates in a shared latent space defined by a Stitch-VAE. The Stitch-VAE is trained jointly across subjects with balanced multi-subject batches and a composite reconstruction and KL-divergence objective.

The stitch VAE is a 2D U-Net VAE trained on individual frames. The encoder uses standard group normalization, while the decoder uses spatially adaptive group-normalization layers [49]. In the default configuration, it uses four resolution stages with encoder channel dimensions $(64, 128, 256, 512)$ and decoder channel dimensions $(512, 256, 128, 64)$, 32 groups for normalization, and ReLU activations. The encoder predicts per-pixel Gaussian parameters $(\mu, \log \sigma^2)$ in a latent feature map with 8 channels.

Let x denote an observed neural response frame. The Stitch-VAE encodes x to a diagonal Gaussian posterior $q_\phi(z \mid x)$ and decodes to a reconstruction $\hat{x}$. The training loss combines a standard variational autoencoder objective [50] with a composite reconstruction term and a KL regularizer,

$$\mathcal{L}_{\text{VAE}}(x) = \mathcal{L}_{\text{px}}(x, \hat{x}) + \beta \, \mathcal{L}_{\text{KL}}\big(q_\phi(z \mid x) \,\|\, p(z)\big), \tag{S6}$$

where $p(z) = \mathcal{N}(0, I)$. The reconstruction term $\mathcal{L}_{\text{px}}$ consists of a weighted multi-scale (pyramid) reconstruction loss and a Sobel edge loss. The weights are set to $w_{\text{pyr}} = 5.0$ and $w_{\text{sobel}} = 1.0$. We warm up the KL weight from $\beta_{\text{start}} = 0$ to $\beta_{\text{end}} = 10^{-3}$ over the first 1000 optimization steps using a linear schedule and keep $\beta(t) = \beta_{\text{end}}$ thereafter. We optimize with AdamW [51] and train for 50 epochs, selecting the checkpoint with the best validation loss.

B.3 State-dependent 3D U-Net latent diffusion

We train a conditional diffusion model to generate future neural responses given (i) recent neural history (state) and (ii) natural images, while conditioning additionally on subject identity.

The denoiser is a 3D U-Net operating on a spatiotemporal latent volume of shape (B, C_z, T, H_z, W_z), where C_z is the Stitch-VAE latent channel count and $T = T_{\text{hist}} + T_{\text{tgt}}$. Convolutions are factorized with kernels of the form $(1, k, k)$, so downsampling and upsampling occur only in space. The network uses residual blocks with timestep embeddings, temporal attention blocks over the frame dimension, and spatial attention (a sparse linear attention variant with a full attention block at the bottleneck). The base width is 512 with spatial channel multipliers $(1, 2, 4)$, 8 attention heads, and attention head dimension 32.

Stimulus conditioning uses a tokenized embedding of dimension 768 per token obtained by encoding the grayscale natural image with the pretrained ViT-MAE visual tokenizer (`https://huggingface.co/facebook/vit-mae-base`) [41]. Each stimulus image is converted to three channels (by channel replication), resized to 224×224, and rescaled to $[0, 1]$ before patch tokenization and encoding. Each token is projected to a 256-dimensional conditioning subspace and passed through a small MLP; token-wise features are then concatenated, yielding a per-frame conditioning vector. Subject identity is represented by a learned 768-dimensional subject embedding, appended as an additional token so that the diffusion model shares weights across subjects while remaining subject-aware.

Each trial is decomposed into overlapping windows of fixed length $T = T_{\text{hist}} + T_{\text{tgt}}$. The dataset enumerates multiple effective history lengths by choosing the target start within the history span, yielding examples with history length $\ell \in \{0, \ldots, T_{\text{hist}}\}$. For each example, the history clip is right-aligned in a tensor of shape T_{hist} with zero-padding for missing history frames.

Diffusion training follows the x_0-prediction objective, applied to the concatenated sequence $[z_{\text{hist}}, z_{\text{tgt}}]$ in which only the target latents are noised. We additionally apply history perturbations during training, including random history dropout with probability 0.1. Optimization uses AdamW with gradient clipping; optionally, an exponential moving average (EMA; decay 0.995) of parameters is maintained for sampling.

B.4 Synthetic data generation

To generate synthetic neural responses, we run guided DDIM sampling in latent space [52] and decode samples with the Stitch-VAE. For each example, we keep the history latents fixed and sample only the target segment by initializing the target latents from Gaussian noise and iterating a DDIM scheduler. For cross-state sampling, conditioning histories are drawn from a global pool of real onset histories pooled across all training stimuli.

B.5 Downstream decoding evaluation

We evaluate the generated synthetic samples on held-out real data by training a 78-way classifier that decodes the image class from the wide-field calcium imaging onset windows. We split real trials into $80\%/10\%/10\%$ train/validation/test, use the validation split for model selection, and report final accuracy on the held-out test split.

To mimic rapid detection of external stimuli, we train and evaluate the decoder using only the first 8 imaging frames after stimulus onset.

Decoder architecture. The decoder is a 3D convolutional residual encoder operating on spatiotemporal neural response volumes. It consists of four 3D residual blocks interleaved with spatial pooling, followed by adaptive pooling and a linear classifier head.

Specifically, the encoder stack is:

- Residual block: kernel $(5, 3, 3)$, stride $(1, 1, 1)$, output channels 128, expansion factor 4, activation TanhShrink, dropout 0.5.
- MaxPool3D: kernel $(1, 7, 7)$, stride $(1, 2, 2)$, padding $(0, 1, 1)$.
- Residual block: kernel $(3, 3, 3)$, stride $(1, 1, 1)$, output channels 64, expansion factor 1, activation LeakyReLU, dropout 0.1.
- MaxPool3D: kernel $(1, 5, 5)$, stride $(1, 1, 1)$, padding $(0, 1, 1)$.
- Residual block: kernel $(1, 7, 7)$, stride $(1, 1, 1)$, output channels 32, expansion factor 4, activation Mish, dropout 0.4.
- MaxPool3D: kernel $(1, 5, 5)$, stride $(1, 2, 2)$, padding $(0, 1, 1)$.
- Residual block: kernel $(3, 3, 3)$, stride $(1, 2, 2)$, output channels 256, expansion factor 4, activation RReLU, dropout 0.4.
- AdaptiveMaxPool3D to output size $(1, 4, 4)$.

The pooled features are flattened and mapped to 78 logits with a linear layer. We optimize with AdamW [51] and train for 100 epochs, selecting the checkpoint with the best validation loss.

Synthetic pretraining and real-data finetuning. For each stimulus class, we generate synthetic neural response clips with our state-dependent latent diffusion model, using the same stimulus labels as in the real dataset. We then train the decoder on this synthetic dataset using a standard cross-entropy objective. After synthetic pretraining, we finetune the same decoder on real training trials and report performance on a held-out real test set.

B.6 State-dependent generative model baselines

We compared BrainStateDiff with five generative baselines on the mouse V1 wide-field calcium dataset: a state-dependent coupled GLM, LFADS [25], pi-VAE [23], Bashiri et al. [30], and LDNS [16]. All comparisons use the first eight post-stimulus response frames (frames 20–27 of each 28-frame 96×96 clip) as the generated target. The common stimulus input is a frozen ViT-MAE image-token representation, denoted $\phi(I)$; each baseline maps this representation into its own conditioning space. The GLM and LFADS additionally condition on the 20-frame pre-stimulus neural history and are evaluated under both matched-state and cross-state sampling. The remaining three baselines do not consume the donor pre-stimulus state and are therefore evaluated only under matched-state sampling.

State-dependent coupled GLM. The GLM is trained separately for each mouse and models raw $\Delta F/F$ image sequences rather than stitch-VAE latents. Each training example contains the 20-frame pre-stimulus/onset history and the eight-frame response target defined above. The GLM maps $\phi(I)$ through a learned low-dimensional projection, denoted $\tilde{\phi}(I)$.

The model treats each wide-field pixel as one response unit and combines three sources of conditioning: the pixel's own recent activity, local spatially coupled activity from nearby pixels, and image

identity. For pixel i and prediction step k, the linear drive is

$$\eta_{i,k} = \beta_i + \sum_{\ell=1}^{L} h^{\text{self}}_{i,\ell} x_{i,k-\ell} + \sum_{j \in \mathcal{N}(i)} \sum_{\ell=1}^{L} h^{\text{cpl}}_{i,j,\ell} x_{j,k-\ell} + \sum_{a} \psi_a(k) \left\langle K_{i,a}, \tilde{\phi}(I) \right\rangle, \tag{S7}$$

where $\mathcal{N}(i)$ is the local 5×5 neighborhood around pixel i with the center pixel removed and $\psi_a(k)$ are raised-cosine temporal basis functions over the prediction window. The response distribution is Gaussian,

$$\mu_{i,k} = b_{i,k} + \alpha_i \,\text{softplus}(\eta_{i,k}), \qquad x_{i,k} \sim \mathcal{N}(\mu_{i,k}, \sigma_i^2). \tag{S8}$$

During generation, the model is rolled out autoregressively so that previously generated frames become the lagged state for subsequent frames.

The GLM is optimized with Gaussian negative log likelihood, Adam learning rate 10^{-3}, batch size 4, gradient clipping at 1.0, L2 penalty 10^{-4}, and group-L1 penalty 10^{-4} over self- and cross-coupling lag filters. Per-pixel, per-frame baselines are initialized from training-set means. Training uses scheduled sampling, with teacher forcing decayed over the first 20 epochs to a floor of 0.2, and checkpoints are selected by autoregressive validation likelihood.

LFADS, pi-VAE, Bashiri et al., and LDNS. The four modern baselines use stitched two-mouse models. For image conditioning, we mean-pool $\phi(I)$ and apply a shared, train-fitted 32-dimensional projection.

LFADS uses an initial-condition encoder, controller, recurrent generator, factor readout, and Gaussian reconstruction distribution. The image condition enters through the canonical `ext_input` pathway, whereas the initial condition is inferred only from the 20-frame pre-stimulus neural history. pi-VAE uses a static label-conditioned Gaussian VAE. Bashiri et al. use an image-conditioned factor-analysis base followed by an eight-bin monotone spline flow. LDNS uses an S4 autoencoder [53] with conditional latent diffusion and a Gaussian observation head.

Hyperparameter search and training. We evaluated neural observation representations and model capacities in stages. The initial screen compared Stitch-VAE latents with pixel-PCA representations containing 16, 24, or 32 components, each at capacity multipliers of $1\times$ and $2\times$. We then evaluated PCA dimensions $\{24, 32, 40, 48\}$ at multipliers $\{1, 2, 4, 8\}$. For pi-VAE, we additionally tested $16\times$ capacity at these dimensions and a 56-component representation at all five capacities. For Bashiri et al., we additionally tested $16\times$ and $32\times$ capacities at PCA dimensions $\{24, 32, 40, 48\}$.

At capacity multiplier c, LFADS uses $100c$ generator units and $50c$ factors. pi-VAE uses $\min(4c^2, 16)$ latent dimensions and an encoder width of $128c$. Bashiri et al. use $4c^2$ factors and two conditioning layers of width $128c$. LDNS uses $64c$ channels in both its autoencoder and denoiser.

Screening runs use 100 epochs and batch size 32, with the epoch budget applied separately to the LDNS autoencoder and diffusion stages. The final LFADS fit uses a maximum of 200 epochs and 16 trials per mouse in each batch. LFADS is optimized with Adam [54], learning rate 4×10^{-3}, and dropout 0.3. The initial-condition and inferred-input KL weights are ramped to 10^{-7} over the first 80 epochs. pi-VAE and Bashiri et al. use Adam, whereas LDNS uses AdamW [51], all with learning rate 10^{-3}.

Within each fixed configuration, LFADS checkpoints minimize validation reconstruction loss, pi-VAE checkpoints minimize validation negative ELBO, and Bashiri et al. checkpoints minimize validation negative log likelihood. LDNS selects its autoencoder and denoiser separately using validation Gaussian reconstruction loss and denoising SmoothL1 loss, respectively. LDNS uses $1{,}000$ diffusion training steps and 100 sampling steps. We selected among configurations using validation synthetic-to-real decoding and report the corresponding held-out test results.

Table S1: **Wide-field trial-average fidelity comparison.** Entries report Pearson correlation between real and synthetic trial-averaged responses for the same image (within-key) and different images (across-key). Mean ± s.e.m. is computed over the two mice.

Method	Sampling	Within-key mean ± s.e.m.	Across-key mean ± s.e.m.
BrainStateDiff	matched-state	$\mathbf{0.946 \pm 0.022}$	-0.025 ± 0.005
BrainStateDiff	cross-state	$\mathbf{0.769 \pm 0.002}$	-0.049 ± 0.004
GLM	matched-state	0.500 ± 0.057	-0.188 ± 0.021
GLM	cross-state	0.464 ± 0.066	-0.192 ± 0.022

Table S2: **Mouse V1 generative-model comparison.** Held-out 78-way classification accuracy. Each cell stacks synthetic-only training over synthetic pretraining followed by real-data finetuning, and columns give the synthetic-to-real scaling factor. Mean ± s.e.m. across $n = 2$ subjects.

Metric	Method	1×	40×	78×
Matched-state				
Accuracy	BrainStateDiff	**0.7072±0.0357** **0.8000±0.0602**	**0.7670±0.0587** **0.8071±0.0820**	**0.7906±0.0466** **0.8250±0.0486**
	LFADS	0.0506±0.0041 0.7910±0.0820	0.2421±0.0419 0.7577±0.0861	0.2476±0.0410 0.7556±0.0800
	GLM	0.0502±0.0116 0.7960±0.0773	0.1181±0.0469 0.7518±0.0777	0.1149±0.0505 0.7590±0.0689
	Bashiri	0.0668±0.0079 0.7885±0.0798	0.1697±0.0505 0.7756±0.1147	0.1702±0.0422 0.7832±0.0786
	pi-VAE	0.0566±0.0070 0.7965±0.0865	0.1359±0.0499 0.7761±0.0868	0.1364±0.0542 0.7556±0.0836
	LDNS	0.0119±0.0009 0.7979±0.0696	0.0143±0.0029 0.7740±0.0824	0.0160±0.0045 0.6944±0.0474
Cross-state				
Accuracy	BrainStateDiff	**0.5774±0.1593** 0.7971±0.0843	**0.7542±0.0718** **0.8429±0.0576**	**0.7638±0.0689** **0.8650±0.0354**
	LFADS	0.0524±0.0011 **0.8003±0.0739**	0.2535±0.0363 0.7489±0.0904	0.2566±0.0446 0.7489±0.0836
	GLM	0.0567±0.0177 0.7984±0.0766	0.0777±0.0111 0.7261±0.0369	0.0742±0.0011 0.7228±0.0616

C THINGS ventral stream spiking dataset of monkey V4

C.1 Data acquisition

We use the THINGS ventral stream spiking dataset (TVSD) [46], which contains multi-unit activity recorded with Utah arrays from two macaques during passive fixation while viewing natural images. Visual stimuli are 500×500 images from THINGS [39] presented in 4-image sequences: each trial begins with 300 ms fixation and then presents four images, where each image is shown for 200 ms followed by 200 ms of a gray screen. The training set comprises $1{,}854$ categories with 12 images per category ($22{,}248$ images) shown once each, and the test set contains 100 images each repeated 30 times.

We focus on V4 recordings and retain electrodes with reliability > 0.3, yielding 315 V4 channels in total (198 from monkey N and 117 from monkey F). For the repeated test images, we average MUA traces across repetitions to obtain a single response per image.

For subsequent modeling, we represent each image-evoked response as a binned time series over a 300 ms window around stimulus onset. The 300 ms window spans 100 ms pre-stimulus to 200 ms post-stimulus onset. Each electrode's MUA signal is normalized and then binned into $T = 30$ temporal bins (10 ms per bin).

Normalization is performed per electrode, per recording day, and per 10 ms temporal segment, the same way as [46, 47]. The full trial time series (sampled at 1 kHz, 1 ms timepoints) is divided into non-overlapping 10 ms windows. For each (electrode, day, 10 ms window) triplet, we pool the MUA values across all trials recorded on that day to compute their mean and standard deviation. Each 1 ms timepoint is then z-scored using these local statistics.

Following normalization, the 1 ms timepoints are averaged within their respective 10 ms intervals to compute the final binned values. To form a unified representation for each trial, the neural responses from both monkeys are concatenated along the electrode dimension. The resulting data for each image presentation is thus structured as a single matrix of shape $T \times N$, where $T = 30$ bins and $N = 315$ total reliable V4 channels (198 from monkey N and 117 from monkey F).

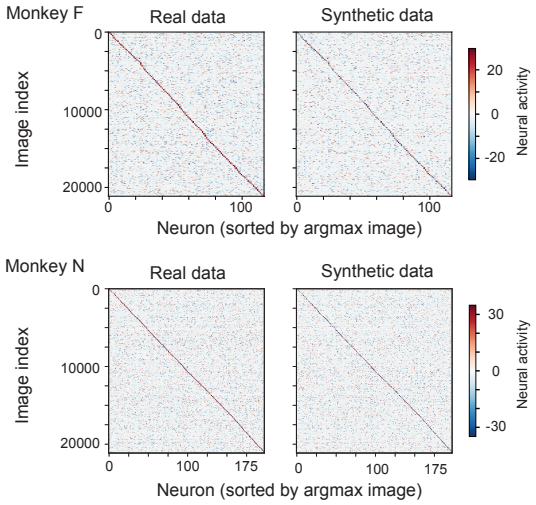

Figure S1: **Representative image-evoked V4 activity for two monkeys.** Rows are 22,248 training images in a fixed order; columns are reliable V4 channels. For each monkey, channels are ordered by the image eliciting their strongest real response, and the same order is applied to generated responses. Preserved diagonal structure indicates retained stimulus-specific population selectivity.

C.2 State-dependent V4 diffusion transformer

We model the conditional distribution of V4 evoked activity given both stimulus identity and pre-stimulus state using a state-dependent diffusion transformer. For each stimulus image, we start from the binned MUA time series (315 channels; $T = 30$ bins) and form two summary vectors: (i) a *baseline state* $\mathbf{s} \in \mathbb{R}^N$ computed as the mean of bins $[0, 10)$, and (ii) an *evoked target* $\mathbf{y} \in \mathbb{R}^N$ computed as the mean of bins $[15, 25)$. This scalar summary mirrors current state-of-the-art V4 decoding algorithms, which use a single time-averaged response vector as decoder input; we therefore model the same scalar evoked target that the downstream decoder consumes. We train a conditional diffusion model to generate $\mathbf{y}$ given the image and baseline state, i.e., $p_\theta(\mathbf{y} \mid \text{image}, \mathbf{s}, \text{subject})$.

Multi-subject handling. The dataset contains two subjects with different numbers of reliable V4 channels (198 for monkey N and 117 for monkey F). We treat each subject as a separate condition via a learned subject embedding and use subject-specific input/output projections. To enable a single shared transformer, we pad neural vectors to `MAX_NEURONS` $= 198$ and use a boolean padding mask to ignore padded entries in attention and loss.

Conditioning and architecture. The diffusion backbone is a transformer operating on neuron tokens (one token per channel). Stimulus conditioning uses frozen tokens from the pretrained ViT-MAE (`https://huggingface.co/facebook/vit-mae-base`) [41] via cross-attention; in addition, we inject global conditioning through AdaLN-style modulation. The global conditioning vector concatenates (a) a timestep embedding, (b) a mean-pooled image embedding, (c) the subject embedding, and (d) a state embedding. The state embedding is computed by a subject-specific MLP that maps the padded baseline vector to a fixed 256-D representation.

Diffusion objective and optimization. Following the x_0-prediction objective, we use a DDIM-style noise schedule [52] with 1,000 training timesteps (betas from 8.5×10^{-4} to 1.2×10^{-2}), with the MSE loss computed only over non-padded channels. We normalize baseline and target components using a subject-wise scalar mean and standard deviation estimated by pooling across neurons and across both components. We optimize with AdamW [51] (learning rate 10^{-4}, weight decay 10^{-4}) with 1,000 warmup steps and maintain an exponential moving average of parameters (decay 0.9999).

C.3 Synthetic data generation

For each subject, we initialize the padded target vector from Gaussian noise and run 50 DDIM inference steps using the trained diffusion transformer to obtain a denoised target $\hat{\mathbf{y}}$. For cross-state sampling, $\mathbf{s}$ is resampled from a pool of baseline states pooled across all training images. We generate multiple synthetic responses per training image to form an augmented (image, neural) dataset for downstream reconstruction.

C.4 Downstream decoding evaluation

Task. We evaluate whether the generated synthetic data using BrainStateDiff faithfully preserve stimulus information through an image reconstruction task, which is the task used in the STSBench [47]. The goal is to reconstruct the presented natural image from V4 neural responses.

Train/validation/test splits. We use the TVSD training set of 22,248 images and apply a fixed $\approx 5\%$ validation split (1,112 images) for checkpoint selection. We evaluate reconstruction quality on the held-out TVSD test set of 100 images (each repeated 30 times in the raw recordings).

Image Reconstruction model. We use a conditional latent diffusion reconstruction model, following the neural image-reconstruction setup used in STSBench and recent latent-diffusion-based brain-to-image pipelines [19, 20, 47, 55]. We first train a VQ-VAE from scratch on the TVSD training images to serve as the image encoder, then freeze it and train the U-Net denoiser on top, following the standard latent-diffusion training order [20]. A U-Net denoiser is then trained to predict diffusion noise in latent space with 1,000 diffusion timesteps (betas from 8.5×10^{-4} to 1.2×10^{-2}). The neural condition is a time-averaged V4 response vector computed over the canonical V4 response window (50–150 ms post-stimulus), using either all 315 channels or a subject-specific subset (198 for monkey N; 117 for monkey F). At test time, the only input to the reconstructor is the held-out neural response. The pre-stimulus state and image tokens condition the generator, not the reconstructor. We apply condition dropout with probability 0.1 during training.

For the synthetic plus real data evaluation, we train the reconstruction model on a mixture of synthetic and real training pairs (mixed training) for 100 epochs, selecting checkpoints using the real validation split. We then report reconstruction performance on the held-out real test images. Aside from the source of training pairs, we keep the reconstruction-model optimization settings fixed across conditions.

Metrics. We report peak signal-to-noise ratio (PSNR; in dB), defined as $\mathrm{PSNR} = 10 \log_{10}(I_{\max}^2/\mathrm{MSE})$ (with $I_{\max} = 1$ for images normalized to $[0, 1]$), and learned perceptual image patch similarity (LPIPS) [56], computed as a learned distance between deep features from an ImageNet-pretrained VGG16 backbone across multiple layers.

C.5 State-dependent generative model baselines

We compared BrainStateDiff with five generative baselines on the monkey V4 dataset: a state-dependent coupled GLM, LFADS [25], pi-VAE [23], Bashiri et al. [30], and LDNS [16]. All comparisons use the mean evoked response over bins 15–24 as the generated target. The common stimulus input is a frozen ViT-MAE image-token representation, denoted $\phi(I)$; each baseline maps this representation into its own conditioning space. The GLM and LFADS additionally condition on the pre-stimulus state, defined as the population-response vector averaged over bins 0–9. Both are evaluated under matched-state and cross-state sampling. The remaining three baselines do not consume the donor pre-stimulus state and are therefore evaluated only under matched-state sampling.

State-dependent coupled GLM. The GLM is trained separately for each monkey and models one scalar response per channel in normalized neural-response space. Its linear drive combines the channel's pre-stimulus activity, full cross-channel couplings, and a learned low-dimensional projection $\tilde{\phi}(I)$ of the shared image representation. For neuron i, the linear drive is

$$\eta_i = \beta_i + h_i s_i + \sum_{j \neq i} W_{ij} s_j + \left\langle K_i, \tilde{\phi}(I) \right\rangle, \tag{S9}$$

where s_i is the normalized pre-stimulus activity of neuron i and h_i is its self-state coefficient. The terms W_{ij} are cross-neuron coupling weights with the diagonal masked out, and K_i contains

neuron-specific image weights. The predicted response is

$$\mu_i = b_i + \alpha_i \,\mathrm{softplus}(\eta_i), \qquad y_i \sim \mathcal{N}(\mu_i, \sigma_i^2), \tag{S10}$$

with neuron-specific offset, gain, and Gaussian variance parameters.

The GLM is optimized with Gaussian negative log likelihood. We use Adam with learning rate 10^{-3} and batch size 4, with gradient clipping at 1.0. The objective includes an L2 penalty of 10^{-4} and an L1 penalty of 10^{-4} on the self- and cross-coupling weights. Per-neuron response offsets are warm-started from the training-set target mean, and checkpoints are selected by validation negative log likelihood with early stopping.

We sample synthetic V4 responses from the trained GLM. For matched-state sampling, each image is paired with its recorded pre-stimulus state. For cross-state sampling, pre-stimulus states are randomly deranged within the selected split so that no image retains its originally observed state.

LFADS, pi-VAE, Bashiri et al., and LDNS. The four modern baselines use stitched models trained jointly across the two monkeys. They use eight-head attention pooling to map all frozen ViT-MAE tokens $\phi(I)$ [41] to a 384-dimensional image condition. LFADS uses a 200-unit recurrent generator, 100 factors, 64-dimensional initial-condition and controller encoders, and four inferred inputs. The image condition enters through the canonical `ext_input` pathway, whereas the initial condition is inferred from the pre-stimulus state. pi-VAE uses a static Gaussian VAE conditioned on the image representation, with four latent dimensions, a 128-unit inference network, and a 32-unit conditional-prior network. Bashiri et al. use an image-conditioned factor-analysis base with four factors, two conditioning layers of width 128, and an eight-bin monotone spline flow.

LDNS uses a two-block, 64-channel S4 autoencoder [53] with eight latent channels and a four-block, 64-channel denoiser for conditional latent diffusion. To represent the static V4 target, each response vector is repeated over eight positions. The decoded means are averaged across these positions before Gaussian observation noise is added.

We use the method-specific optimization settings described in Appendix B.6, with 200 epochs per model or LDNS stage. pi-VAE, Bashiri et al., and LDNS use batch size 256; each LFADS update uses 256 examples per monkey. Generator checkpoints are selected using the corresponding validation objectives described in Appendix B.6. For V4, LDNS uses $1{,}000$ diffusion training steps and $1{,}000$ sampling steps. Downstream image-reconstruction checkpoints are selected separately for each monkey, scaling factor, and training regime using validation PSNR or LPIPS [56]. The corresponding held-out test values are reported.

Table S3: **Monkey V4 generative-model comparison.** Held-out image-reconstruction performance measured by PSNR (dB; ↑) and LPIPS (↓). Each cell stacks synthetic-only training over mixed training with synthetic and real pairs, and columns give the synthetic-to-real scaling factor. Mean ± s.e.m. across $n = 2$ subjects.

Metric	Method	1×	2×	5×	10×
Matched-state					
PSNR ↑	BrainStateDiff	**10.3702±0.1023** **10.4087±0.1401**	**10.5858±0.0202** **10.6755±0.1683**	**10.7587±0.2462** **10.8936±0.0706**	**11.0141±0.1699** **10.9037±0.0502**
	LFADS	9.7646±0.1403 9.9334±0.1703	9.9070±0.2436 10.2025±0.1408	10.2411±0.0518 10.3239±0.0899	10.5593±0.0269 10.6164±0.0039
	GLM	9.9409±0.2499 9.9662±0.1531	9.8789±0.3119 10.1271±0.1354	10.2668±0.1462 10.2513±0.1407	10.6895±0.1051 10.5567±0.1298
	Bashiri	9.8236±0.0954 10.0121±0.0275	9.8617±0.1090 10.3499±0.0446	10.4550±0.1028 10.3950±0.0792	10.7428±0.0040 10.6768±0.1405
	pi-VAE	9.6425±0.2112 9.8113±0.0840	9.6084±0.1610 9.8483±0.0942	10.1318±0.1664 10.0934±0.0704	10.2967±0.0769 10.3636±0.2873
	LDNS	9.5043±0.2225 9.9391±0.0113	9.5961±0.2368 9.8871±0.1250	10.0152±0.0279 10.0304±0.0629	10.2864±0.0604 10.1968±0.0912
LPIPS ↓	BrainStateDiff	**0.7344±0.0009** **0.7350±0.0006**	**0.7290±0.0056** **0.7343±0.0003**	**0.7290±0.0016** **0.7265±0.0020**	**0.7290±0.0043** **0.7312±0.0010**
	LFADS	0.7514±0.0044 0.7395±0.0005	0.7439±0.0020 0.7414±0.0047	0.7480±0.0034 0.7458±0.0006	0.7443±0.0030 0.7448±0.0014
	GLM	0.7570±0.0006 0.7473±0.0005	0.7536±0.0014 0.7514±0.0027	0.7503±0.0008 0.7504±0.0056	0.7463±0.0055 0.7486±0.0039
	Bashiri	0.7455±0.0048 0.7409±0.0003	0.7389±0.0011 0.7409±0.0007	0.7438±0.0018 0.7392±0.0015	0.7400±0.0053 0.7376±0.0004
	pi-VAE	0.7586±0.0017 0.7514±0.0072	0.7555±0.0008 0.7532±0.0028	0.7583±0.0012 0.7534±0.0019	0.7540±0.0025 0.7565±0.0025
	LDNS	0.7637±0.0015 0.7485±0.0047	0.7618±0.0040 0.7512±0.0020	0.7607±0.0033 0.7575±0.0012	0.7591±0.0024 0.7554±0.0053
Cross-state					
PSNR ↑	BrainStateDiff	**10.2288±0.1027** **10.3219±0.0673**	**10.5491±0.0100** **10.6718±0.0058**	**10.8095±0.2164** **10.9777±0.1716**	**11.0235±0.1544** **11.0768±0.0609**
	LFADS	9.5870±0.2095 9.8113±0.0435	9.9635±0.1060 10.0728±0.0395	10.1797±0.1944 10.2811±0.1190	10.4908±0.1422 10.6146±0.1505
	GLM	9.8641±0.3564 10.0861±0.0352	9.9182±0.1279 10.0973±0.0147	10.4154±0.1464 10.2250±0.1132	10.5634±0.2586 10.6899±0.2397
LPIPS ↓	BrainStateDiff	**0.7294±0.0017** **0.7292±0.0003**	**0.7332±0.0030** **0.7344±0.0026**	**0.7296±0.0012** **0.7336±0.0056**	**0.7289±0.0002** **0.7307±0.0035**
	LFADS	0.7501±0.0027 0.7429±0.0002	0.7456±0.0045 0.7465±0.0067	0.7450±0.0027 0.7471±0.0056	0.7458±0.0031 0.7393±0.0010
	GLM	0.7599±0.0006 0.7513±0.0011	0.7535±0.0010 0.7508±0.0024	0.7490±0.0048 0.7514±0.0054	0.7479±0.0050 0.7474±0.0021

D Two-photon calcium imaging dataset of mouse dorsal-lateral striatum

D.1 Data acquisition

We use a two-photon calcium imaging dataset from the mouse dorsal-lateral striatum collected across 13 recording sessions from 3 subjects [33]. Each session contains variable-length trials with neural activity vector (∼200 neurons per session), wheel velocity as behavior (v_t), cursor position as stimulus (c_t), and discrete cue label (`Cue-Left` or `Cue-Right`) indicating which side of the screen the initial cursor stimulus appears.

For subsequent modeling, we represent each trial as a sequence of neural frames and behavior aligned in time. We define an anchor frame at stimulus onset and use the preceding $H = 16$ frames as the state. We split real trials into $6 : 3 : 1$ train/validation/test sets.

D.2 Stitch-VAE

To obtain a compact and session-aligned neural representation, we train a Stitch-VAE. Because the number of simultaneously imaged neurons can vary across sessions, this model uses session-specific input/output layers (1D convolution layer) while sharing the encoder/decoder backbone across all sessions, thereby learning a common latent space.

Architecture. For a session with N_s neurons, the VAE encodes each timepoint independently: each neural frame $\mathbf{x}_t \in \mathbb{R}^{N_s}$ is mapped to a latent $\mathbf{z}_t \in \mathbb{R}^D$. A session-specific input layer projects the N_s-dimensional frame into a shared encoder composed of ResNet-style 1D blocks with optional strided downsampling, followed by a bottleneck convolution that outputs $2D$ channels to parameterize a diagonal Gaussian posterior $q_\phi(\mathbf{z}_t \mid \mathbf{x}_t)$ via $(\boldsymbol{\mu}, \log \boldsymbol{\sigma}^2)$. The shared decoder mirrors the encoder.

Training objective and protocol. The Stitch-VAE is trained jointly across sessions. For padded variable-length trials, we construct a per-timepoint mask from each trial's length and compute the reconstruction loss only on valid timepoints. Training minimizes a standard VAE objective [50]

$$\mathcal{L}_{\mathrm{VAE}} = \mathcal{L}_{\mathrm{recon}} + \beta\,\mathrm{KL}\big(q_\phi(\mathbf{z} \mid \mathbf{x}) \,\|\, \mathcal{N}(0, I)\big),$$

where we use a Gaussian negative log-likelihood reconstruction loss with a predicted per-neuron log-variance. We optimize with Adam (learning rate 10^{-4}, weight decay 10^{-4}) with gradient clipping (5.0) and select the best checkpoint by the weighted average validation loss across sessions. In our experiments, we use $D = 16$, 256 base channels, channel multipliers of 1 and 2 for the two encoding layers, dropout 0.3, KL weight 10^{-6} (linearly warmed up over 100 epochs).

D.3 Chunk-level autoregressive latent diffusion

We model the conditional distribution of future neural dynamics and behavior given the recent state using a chunk-level autoregressive conditional diffusion model.

Chunked representation. For a given anchor frame t, we form a history prefix

$$(\mathbf{z}_{t-H:t-1},\ \mathbf{v}_{t-H:t-1},\ \mathbf{c}_{t-H:t-1}),$$

where $H = 16$. The future suffix starting at t is partitioned into non-overlapping chunks of length $\kappa = 8$. For chunk index i we define the diffusion target

$$\mathbf{y}^{(i)} = [\mathbf{z}_{t+i\kappa:t+(i+1)\kappa-1},\ \mathbf{v}_{t+i\kappa:t+(i+1)\kappa-1}] \in \mathbb{R}^{\kappa\times(D+1)}.$$

The cursor is treated as a conditioning stimulus signal rather than a diffusion target. For training, we use a sliding-anchor dataset that enumerates anchors $t \in [16, T-\kappa]$ with stride 1, yielding multiple samples per trial; for sampling and evaluation, we use a fixed anchor at frame 16 (stimulus onset). The final chunk is zero-padded and accompanied by a validity mask.

Normalization. All model inputs/outputs operate in normalized space using global mean/standard deviation computed from training frames, pooled across sessions for $\mathbf{z}$, v, and c.

Denoiser architecture and conditioning. The denoiser is a diffusion transformer trained with diffusion v-prediction. A condition is built from the history by projecting per-frame concatenations $[\mathbf{z}_t, v_t, c_t]$ to a model width of 256, adding learned positional embeddings, and appending a learned trial-type token. Noisy future chunk tokens $\mathbf{y}_t$ are projected to the same width and processed by a 6-layer decoder with self-attention, cross-attention to the condition, and an MLP in each block. Global conditioning is injected via AdaLayerNorm using the sum of (i) a sinusoidal diffusion-timestep embedding, (ii) a learned identity embedding with separate subject and session components, and (iii) a sinusoidal embedding of the chunk offset (anchor shift relative to stimulus onset).

Diffusion objective and optimization. We use a DDIM-style scheduler [52] with 1,000 training timesteps and a squared-cosine beta schedule, and train with v-prediction. The training loss is a masked MSE over valid future frames, split across latent and velocity channels and combined as $\mathcal{L}_z + \mathcal{L}_v$. We optimize with AdamW [51] (learning rate 10^{-4}, weight decay 10^{-4}) for 200 epochs with gradient clipping (1.0) and an optional cosine learning-rate schedule with warmup.

Mixed teacher forcing and rollout training. To reduce drift during closed-loop generation, each sample is stochastically assigned to teacher forcing or rollout, following scheduled-sampling style training [57, 58]. The teacher-forcing probability decays over epochs as $p_{\mathrm{TF}} = \max(p_{\min},\ p_0\,\gamma^{\mathrm{epoch}})$ with defaults $p_0 = 1.0$, $\gamma = 0.95$, and $p_{\min} = 0.2$. During rollout updates, the next history chunk is sampled from the model via short DDIM sampling of 10 steps [52]; cursor history for the generated chunk is recomputed from generated velocity using a session-specific linear dynamics rule that matches the experimental task dynamics.

D.4 Synthetic data generation

We generate synthetic striatal trials using closed-loop autoregressive rollouts in the stitched latent space. Each rollout is initialized from a real history state consisting of the previous $H = 16$ frames of neural latent activity, wheel velocity, and cursor position, together with the cue label and subject/session identity. For matched-state sampling, this history is taken from the same real trials.

For cross-state sampling, the cue and recording identity are fixed, but the initial history is drawn from a within-session pool of trial histories pooled across cue types, thereby recombining observed behavioral/neural states with different trial cues.

At rollout step i, the current history state conditions the diffusion transformer, which samples the next chunk

$$\hat{\mathbf{y}}^{(i)} = [\hat{\mathbf{z}}_{i\kappa:(i+1)\kappa-1}, \hat{\mathbf{v}}_{i\kappa:(i+1)\kappa-1}]$$

using DDIM sampling with 50 inference steps. The generated wheel velocity is then integrated through the session-specific linear cursor dynamics to obtain the corresponding cursor positions. We update the conditioning history by appending the generated latent activity, wheel velocity, and recomputed cursor trajectory, dropping the oldest κ frames, and repeating this process to produce a variable-length synthetic trial.

Rollout continues until one of three stopping conditions is met: (i) success, if any generated cursor position falls within a threshold of 0.2 from the target center; (ii) wrong-direction termination, if the cursor moves away from the cue-indicated target direction for two consecutive chunks; or (iii) a maximum rollout length of 18 chunks. These hyperparameters match the experimental acceptance criteria for successful task completion. For robustness, we retry generation up to 5 times for trials that terminate due to wrong-direction failure, and discard trials that repeatedly fail. In practice, most rollouts reach the target within the maximum length, and failure modes are rare.

For downstream behavioral decoding, generated latent trajectories are mapped back to session-specific neural activity using the Stitch-VAE decoder. We then construct synthetic training examples from the generated neural activity and wheel velocity using the same frame-wise forecasting targets for held-out real trials evaluation.

D.5 Downstream decoding evaluation

We evaluate whether synthetic data faithfully preserve neural–behavioral dynamics information on three behavioral regression tasks of increasing difficulty: (1) instantaneous decoding of frame-wise wheel velocity from the current neural activity, i.e., predict v_t from $\hat{\mathbf{x}}_t$; (2) short-horizon forecasting, predict v_{t+3} from $\hat{\mathbf{x}}_t$; and (3) longer-horizon forecasting, predict v_{t+7} from $\hat{\mathbf{x}}_t$.

We evaluate the linear decoder under a synthetic-pretraining and real-data finetuning protocol: the decoder is first trained on synthetic samples and then finetuned on real training trials. Performance is reported as the coefficient of determination R^2 on the held-out real test split, averaged over 5 random seeds, and compared to a baseline decoder trained on real data only.

D.6 State-dependent generative model baselines

We compared BrainStateDiff with five generative baselines on the mouse striatum dataset: a state-dependent coupled GLM, LFADS [25], pi-VAE [23], Bashiri et al. [30], and LDNS [16]. All methods receive the binary cue as the external task condition and are evaluated using the wheel-velocity decoding protocol described above. Paper horizons $K = 0, 3, 7$ correspond to decoder offsets 1, 4, and 8. Each session value is averaged over decoder seeds 12345–12349, and summary values are mean $\pm$ SD across $n = 13$ sessions. BrainStateDiff and the stitched V2 pipelines generate joint future neural, cursor-position, and wheel-velocity trajectories. The GLM instead generates future neural activity conditioned on the real future-velocity trace, which is reused as the decoder label. BrainStateDiff, LFADS, and the GLM consume the 16-frame pre-stimulus history and are evaluated under both matched-state and cross-state sampling. pi-VAE, Bashiri et al., and LDNS do not consume the donor pre-stimulus state and are therefore evaluated only under matched-state sampling. BrainStateDiff and the GLM operate in stitched-VAE-reconstructed neural space, whereas the stitched V2 baselines operate on raw $\Delta F/F$ activity.

State-dependent coupled GLM. The GLM is trained separately for each recording session. For every real trial, we pass each neural frame through the frozen session-bound stitched VAE encoder and decoder, yielding reconstructed neural activity $\tilde{\mathbf{x}}_t \in \mathbb{R}^{N_s}$ for session s. The GLM receives a recent history window of length $H = 16$, the corresponding history of wheel velocity, future wheel velocity, and the binary cue variable $q \in \{-1, +1\}$. Variable-length future segments are padded to the maximum training-trial length within the session and masked in the likelihood.

For neuron i and future step k, the GLM combines self-history filters, full pairwise coupling filters, cue-dependent residual filters, wheel-velocity filters, and cue-by-velocity interactions. Let $\psi_a(k)$ denote a raised-cosine basis over future time, and let $v_{k,l}$ denote the wheel velocity at lag l relative to future step k. With the diagonal of the coupling filters masked out, the linear drive is

$$\eta_{i,k} = \beta_i + \sum_l h^0_{i,l}\tilde{x}_{i,k-l} + \sum_{j\neq i}\sum_l H^0_{i,j,l}\tilde{x}_{j,k-l} + q\left(\sum_l h^q_{i,l}\tilde{x}_{i,k-l} + \sum_{j\neq i}\sum_l H^q_{i,j,l}\tilde{x}_{j,k-l}\right) + q\sum_a \psi_a(k)\rho_{i,a} + \sum_a\sum_l \psi_a(k)\theta_{i,a,l}v_{k,l} + q\sum_a\sum_l \psi_a(k)\lambda_{i,a,l}v_{k,l}. \tag{S11}$$

The predicted response mean is

$$\mu_{i,k} = b_{i,k} + \alpha_i \,\mathrm{softplus}(\eta_{i,k}), \qquad \tilde{x}_{i,k} \sim \mathcal{N}(\mu_{i,k}, \sigma_i^2), \tag{S12}$$

with per-neuron observation variance. During autoregressive generation, previously generated means are fed back as neural history inputs; synthetic trials are then obtained by adding Gaussian observation noise using the learned σ_i^2.

The GLM is optimized with a masked Gaussian negative log likelihood over valid future frames. We use Adam with learning rate 10^{-3}, batch size 16, gradient clipping at 1.0, L2 penalty 10^{-4}, group-L1 penalty 10^{-4} on self and coupling lag filters, and early stopping based on autoregressive validation negative log likelihood. The per-frame baseline $b_{i,k}$ is initialized from the training-set mean response. Training uses scheduled sampling: the probability of teacher forcing decays exponentially over the first 20 epochs to a floor of 0.2, and validation checkpoint selection is performed in fully autoregressive mode.

Synthetic sampling. For matched-state sampling, each synthetic copy uses the same trial history, future wheel-velocity trace, and cue as the real training trials. For cross-state sampling, the future wheel-velocity trace is kept from the target trial, while the neural and behavioral history is drawn from a different donor trial in the same session. Cue labels are balanced across the synthetic seeds, with half of the generated copies using $q = +1$ and half using $q = -1$.

LFADS, pi-VAE, Bashiri et al., and LDNS. The four modern baselines use stitched V2 models. LFADS encodes the 16-frame pre-stimulus history with a variational initial-condition encoder. It also uses a controller with an autoregressive inferred-input prior, a recurrent generator, a factor readout, and Gaussian reconstruction. Its initial-condition, inferred-input, generator, and factor dimensions are 64, 4, 200, and 100, respectively.

pi-VAE is a static, per-timepoint Gaussian VAE with a cue-conditioned prior, latent dimension 4, and encoder and label-prior hidden dimensions of 128 and 32. The Bashiri et al. model is static and per-timepoint. It uses a two-layer 128-unit stimulus MLP, a four-factor Gaussian base distribution, and an eight-bin monotone spline flow with tail bound 5. LDNS uses a two-block, 64-channel S4 autoencoder [53] with eight latent channels and a four-block, 64-channel cue-conditioned DDPM [19].

Training and sampling. LFADS is optimized with Adam [54], learning rate 4×10^{-3}, dropout 0.3, and 200 training epochs. Session-specific input mappings project joint neural and behavioral observations to a common width of 131. Each batch contains 76 trials per session, giving 988 trials across the 13 sessions. The reconstruction window includes the 16-frame history and up to 32 future frames.

pi-VAE and Bashiri et al. are optimized with Adam, learning rate 10^{-3}, batch size 256, and 200 training epochs. Their checkpoints minimize validation negative ELBO and negative log likelihood, respectively, with equal weighting across sessions.

LDNS uses Adam with the same learning rate and batch size, and 200 epochs separately for its autoencoder and diffusion stages. The autoencoder is selected using masked Gaussian reconstruction likelihood, and the diffusion model is selected using masked SmoothL1 noise-prediction loss. Variable-length future sequences are center-padded to 138 frames, with padded positions excluded from both objectives. LDNS uses 1,000 diffusion training steps and 100 sampling steps.

During generation, pi-VAE and Bashiri et al. sample future timepoints independently conditional on the cue, whereas LDNS generates the future sequence jointly.

Table S4: **Mouse-striatum generative-model comparison.** Held-out wheel-velocity decoding performance measured by R^2 at horizons $K = 0, 3, 7$ under matched-state sampling; cross-state results continue in the following panel. Each cell stacks synthetic-only training over synthetic pretraining followed by real-data finetuning, and columns give the synthetic-to-real scaling factor. Mean $\pm$ SD across $n = 13$ sessions after averaging five decoder seeds per session.

Horizon	Method	1×	5×	11×
Matched-state				
$K = 0$	BrainStateDiff	**0.7417±0.0494** 0.7645±0.0489	**0.7662±0.0498** 0.7735±0.0503	**0.7752±0.0477** **0.7792±0.0492**
	LFADS	0.6793±0.0599 0.7714±0.0349	0.7136±0.0564 0.7719±0.0364	0.7183±0.0546 0.7737±0.0357
	GLM	0.6105±0.2091 0.7723±0.0353	0.6449±0.1719 0.7746±0.0348	0.6574±0.1454 0.7745±0.0364
	Bashiri	0.6759±0.0535 0.7691±0.0339	0.6955±0.0684 0.7676±0.0324	0.6968±0.0777 0.7733±0.0355
	pi-VAE	0.7351±0.0370 **0.7739±0.0344**	0.7598±0.0401 **0.7779±0.0347**	0.7624±0.0381 0.7772±0.0370
	LDNS	0.6274±0.0583 0.7702±0.0349	0.6659±0.0496 0.7738±0.0329	0.6717±0.0502 0.7733±0.0335
$K = 3$	BrainStateDiff	**0.6633±0.0501** 0.6947±0.0471	**0.6966±0.0491** 0.7060±0.0473	**0.7026±0.0487** 0.7114±0.0459
	LFADS	0.6033±0.0574 **0.7139±0.0471**	0.6500±0.0673 0.7059±0.0767	0.6606±0.0703 0.7083±0.0819
	GLM	0.5398±0.1656 0.7134±0.0489	0.5827±0.1326 0.7162±0.0470	0.5881±0.1159 0.6928±0.0496
	Bashiri	0.5282±0.1314 0.7131±0.0479	0.5421±0.1407 **0.7184±0.0448**	0.5537±0.1424 0.7134±0.0614
	pi-VAE	0.4717±0.0827 0.7111±0.0451	0.5034±0.0868 0.7151±0.0454	0.5105±0.0865 0.7169±0.0458
	LDNS	0.4974±0.0665 0.7122±0.0498	0.5360±0.0611 0.7175±0.0507	0.5405±0.0574 **0.7187±0.0504**
$K = 7$	BrainStateDiff	**0.4426±0.0491** 0.4573±0.0554	**0.4741±0.0541** 0.4795±0.0531	**0.4781±0.0575** 0.4845±0.0528
	LFADS	0.3493±0.0819 0.4768±0.0712	0.3897±0.0973 0.4744±0.0837	0.4104±0.1090 0.4670±0.1247
	GLM	0.2937±0.1010 0.4759±0.0621	0.3385±0.0817 0.4794±0.0602	0.3474±0.0743 0.4545±0.0627
	Bashiri	0.3047±0.1234 0.4652±0.1184	0.3220±0.1657 **0.4862±0.0803**	0.3285±0.1755 **0.4907±0.0866**
	pi-VAE	0.2771±0.0716 **0.4771±0.0590**	0.3188±0.0779 0.4840±0.0584	0.3278±0.0778 0.4841±0.0592
	LDNS	0.2469±0.0759 0.4760±0.0630	0.2934±0.0717 0.4813±0.0645	0.3018±0.0667 0.4841±0.0637

Table S4: **Mouse-striatum generative-model comparison (continued): cross-state sampling.** Entries follow the matched-state panel; only BrainStateDiff, LFADS, and GLM are shown.

Horizon	Method	$1\times$	$5\times$	$11\times$
Cross-state				
$K = 0$	BrainStateDiff	$\mathbf{0.6730 \pm 0.0440}$	$\mathbf{0.7657 \pm 0.0530}$	$\mathbf{0.7780 \pm 0.0481}$
	LFADS	0.7680 ± 0.0350	0.7720 ± 0.0498	$\mathbf{0.7831 \pm 0.0475}$
		0.6700 ± 0.0610	0.7119 ± 0.0563	0.7222 ± 0.0531
	GLM	$\mathbf{0.7700 \pm 0.0346}$	$\mathbf{0.7731 \pm 0.0367}$	0.7745 ± 0.0355
		-0.1518 ± 0.0990	0.3805 ± 0.2066	0.4292 ± 0.1954
		0.7564 ± 0.0330	0.7639 ± 0.0388	0.7651 ± 0.0390
$K = 3$	BrainStateDiff	$\mathbf{0.6080 \pm 0.0630}$	$\mathbf{0.7012 \pm 0.0477}$	$\mathbf{0.7085 \pm 0.0490}$
	LFADS	0.7110 ± 0.0490	0.7093 ± 0.0468	$\mathbf{0.7144 \pm 0.0490}$
		0.5849 ± 0.0702	0.6492 ± 0.0620	0.6625 ± 0.0646
	GLM	$\mathbf{0.7129 \pm 0.0474}$	$\mathbf{0.7094 \pm 0.0667}$	0.7104 ± 0.0729
		-0.1299 ± 0.0566	0.2735 ± 0.1403	0.3360 ± 0.1372
		0.6928 ± 0.0496	0.7014 ± 0.0548	0.7045 ± 0.0545
$K = 7$	BrainStateDiff	$\mathbf{0.3730 \pm 0.0880}$	$\mathbf{0.4803 \pm 0.0535}$	$\mathbf{0.4874 \pm 0.0560}$
	LFADS	$\mathbf{0.4780 \pm 0.0720}$	$\mathbf{0.4842 \pm 0.0541}$	$\mathbf{0.4893 \pm 0.0551}$
		0.3184 ± 0.0984	0.3841 ± 0.1027	0.4033 ± 0.1035
	GLM	0.4774 ± 0.0701	0.4689 ± 0.0986	0.4785 ± 0.0866
		-0.1260 ± 0.0304	0.0926 ± 0.0622	0.1468 ± 0.0642
		0.4545 ± 0.0612	0.4728 ± 0.0611	0.4761 ± 0.0649

The stitched V2 rows use the common baseline harness, whereas BrainStateDiff uses its own epoch-50 generation harness and the GLM uses its V1 pipeline. Cross-method differences involving BrainStateDiff or the GLM are therefore descriptive rather than controlled same-harness comparisons. For the displayed LFADS and GLM cross-state rows, $1\times$ is a per-trial counterfactual flip, $5\times$ is pure block-balanced cross-state sampling, and $11\times$ combines $1\times$ matched-state data with $10\times$ cross-state data. BrainStateDiff uses its own cross-state sampler.

E Resources for reproducibility

All experiments were run on institutional compute clusters. The wide-field calcium decoding analysis requires GPUs with 80 GB of memory. For the remaining computational tasks, including diffusion-model training, synthetic data generation, and downstream analyses on the V4 and striatal datasets, GPUs with at least 24 GB of memory are recommended.

F Mutual information between pre-stimulus state and stimulus

To test independence assumption, we estimated the mutual information $I(\mathbf{S}; U \mid M)$ between pre-stimulus state and stimulus for each recording against a within-session permutation null (1,000 shuffles).

Table S5: **State–stimulus mutual-information diagnostic.** Entries report MI minus the within-session permutation-null control in bits (mean ± SEM); parentheses give recordings with $p < 0.05$ over the total tested.

Dataset	Pre-stimulus	Evoked (positive control)
Mouse V1	-0.003 ± 0.003 (0/2)	0.392 ± 0.106 (2/2)
Monkey V4	$+0.001 \pm 0.002$ (0/2)	0.941 ± 0.155 (2/2)
Mouse striatum (initial state only)	-0.007 ± 0.007 (0/13)	0.713 ± 0.053 (13/13)

Across all datasets, pre-stimulus MI was indistinguishable from zero ($I \leq 0.001$ bits, no recording was significant). In contrast, evoked-window MI was 0.4–0.9 bits and significant in every recording. The measured pre-stimulus state thus carries no detectable information about the upcoming stimulus, supporting the factorization in Eq. 7.

G Held-out true-state versus different-state validation

For each held-out trial, we use the same BrainStateDiff model to generate two responses to the same stimulus: one conditioned on the trial's true pre-stimulus state, one on a state from a different trial. Holding the stimulus fixed isolates the state's contribution—if the model has learned genuine state dependence, the true state should predict the recorded response better. We report energy score and RMSE and correlation. Here Δ is true minus different-state. We compare against LFADS on the mouse V1, monkey V4, and mouse striatum data (Tables S6–S8; mean ± SD across subjects).

Table S6: **True-state vs. sampled-state prediction on held-out mouse V1 trials.**

Metric	Method	True state	Different state	Δ
Energy score ↓	**BrainStateDiff**	13.735 ± 0.307	15.705 ± 0.088	$\mathbf{-1.970 \pm 0.219}$
Energy score ↓	LFADS	15.595 ± 0.013	15.882 ± 0.001	-0.287 ± 0.012
RMSE ↓	**BrainStateDiff**	0.847 ± 0.021	0.977 ± 0.006	$\mathbf{-0.130 \pm 0.016}$
RMSE ↓	LFADS	0.955 ± 0.001	0.969 ± 0.001	-0.015 ± 0.001
Correlation ↑	**BrainStateDiff**	0.514 ± 0.039	0.192 ± 0.033	$\mathbf{+0.322 \pm 0.006}$
Correlation ↑	LFADS	0.248 ± 0.010	0.188 ± 0.008	$+0.060 \pm 0.001$

Table S7: **True-state vs. sampled-state prediction on held-out monkey V4 trials.**

Metric	Method	True state	Different state	Δ
Energy score ↓	**BrainStateDiff**	7.331 ± 0.556	8.175 ± 0.950	$\mathbf{-0.844 \pm 0.394}$
Energy score ↓	LFADS	7.230 ± 1.035	7.573 ± 1.245	-0.343 ± 0.210
RMSE ↓	**BrainStateDiff**	0.766 ± 0.053	0.838 ± 0.030	$\mathbf{-0.071 \pm 0.023}$
RMSE ↓	LFADS	0.835 ± 0.035	0.870 ± 0.019	-0.035 ± 0.016
Correlation ↑	**BrainStateDiff**	0.586 ± 0.030	0.529 ± 0.013	$\mathbf{+0.056 \pm 0.016}$
Correlation ↑	LFADS	0.461 ± 0.011	0.428 ± 0.002	$+0.033 \pm 0.013$

Table S8: **True-state vs. different-state prediction on held-out mouse-striatum trials.**

Metric	Method	True state	Different state	Δ
Energy score ↓	**BrainStateDiff**	2.637 ± 0.659	5.008 ± 1.357	$\mathbf{-2.372 \pm 1.106}$
Energy score ↓	LFADS	5.352 ± 0.838	7.354 ± 2.036	-2.001 ± 1.597
RMSE ↓	**BrainStateDiff**	0.291 ± 0.053	0.478 ± 0.085	$\mathbf{-0.187 \pm 0.076}$
RMSE ↓	LFADS	0.499 ± 0.070	0.641 ± 0.118	-0.142 ± 0.099
Correlation ↑	**BrainStateDiff**	0.745 ± 0.083	0.241 ± 0.112	$\mathbf{+0.504 \pm 0.133}$
Correlation ↑	LFADS	0.345 ± 0.065	0.161 ± 0.064	$+0.184 \pm 0.052$

As shown in the three tables, both BrainStateDiff and LFADS benefit from the true state on held-out trials. BrainStateDiff shows a consistently larger numerical improvement across energy score, RMSE, and response correlation, indicating that its generated responses depend strongly on trial-specific brain state. Together, these tests provide direct evidence that BrainStateDiff uses brain state in a biologically meaningful way.

H Per-animal augmentation gains

We calculated the effects separately for each animal via a paired Wilcoxon signed rank test (Tables S9 and S10).

Table S9: **Per-animal monkey V4 reconstruction gains.** Significance was assessed using a paired Wilcoxon signed rank test.

Metric	Animal	Matched 5×	Matched 10×	Cross 5×	Cross 10×
PSNR	monkeyN	$+0.72^{***}$	$+0.81^{***}$	$+0.60^{***}$	$+0.63^{***}$
	monkeyF	$+0.23^{\mathrm{ns}}$	$+0.64^{***}$	$+0.55^{***}$	$+0.69^{***}$
LPIPS	monkeyN	$+0.009^{*}$	$+0.012^{*}$	$+0.006^{\mathrm{ns}}$	$+0.010^{***}$
	monkeyF	$+0.014^{***}$	$+0.016^{***}$	$+0.010^{***}$	$+0.012^{***}$

$^{***}p < 10^{-4}$, $^{**}p < 0.01$, $^{*}p < 0.05$; ns, non-significant.

Table S10: **Per-animal mouse V1 decoding gains under finetuning.** Significance was assessed using a paired Wilcoxon signed rank test.

Animal	Matched 78×	Cross 78×
mouse1	$+0.024^{*}$	$+0.032^{**}$
mouse2	$+0.056^{***}$	$+0.093^{***}$

$^{***}p < 10^{-4}$, $^{**}p < 0.01$, $^{*}p < 0.05$; ns, non-significant.

I Direct response-space coverage

Starting from a common real-data base A, we add either held-out real responses B, matched-state generated responses, or cross-state generated responses, and quantify coverage expansion using two complementary metrics. Covariance log-determinant gain,

$$\Delta_{\log\det}(X) = \log\det \Sigma_{A\cup X} - \log\det \Sigma_A, \tag{S13}$$

measures the change in the overall volume occupied by the response distribution [59]. Participation-ratio gain,

$$\Delta_{\mathrm{PR}}(X) = \mathrm{PR}(A \cup X) - \mathrm{PR}(A), \qquad \mathrm{PR}(Z) = \frac{\left(\sum_j \lambda_j\right)^2}{\sum_j \lambda_j^2}, \tag{S14}$$

measures how many distinct response directions are meaningfully occupied [60], where λ_j are the covariance eigenvalues. Each parenthetical normalized gain is the generated-response gain divided

by the held-out-real gain. A normalized gain should be near 1, meaning that generated responses change the empirical geometry by approximately the same amount as held-out real responses.

For OEG, gains were computed per stimulus in a 10-dimensional response basis; LFADS used the pixel-PCA 40 configuration.

Table S11: **Direct response-space coverage on OEG.** Per-stimulus, $d = 10$, with LFADS pixel-PCA 40; parentheses report gain relative to adding held-out real responses.

Metric	Method or reference	Real-response Δ	Matched-state Δ	Cross-state Δ
Log det	Real responses	$+5.7102 \pm 0.1896$	–	–
	BrainStateDiff	–	$\mathbf{+4.1864 \pm 0.2949}$ $\mathbf{(0.73\times)}$	$\mathbf{+4.2092 \pm 0.3048}$ $\mathbf{(0.74\times)}$
	LFADS	–	$+1.2167 \pm 0.0908$ $(0.21\times)$	$+1.1807 \pm 0.0898$ $(0.21\times)$
Participation ratio	Real responses	$+0.9903 \pm 0.0069$	–	–
	BrainStateDiff	–	$\mathbf{+1.0456 \pm 0.0200}$ $\mathbf{(1.06\times)}$	$\mathbf{+1.0392 \pm 0.0091}$ $\mathbf{(1.05\times)}$
	LFADS	–	$+0.6391 \pm 0.0002$ $(0.65\times)$	$+0.6281 \pm 0.0175$ $(0.63\times)$

For TVSD, the real responses were split into per-stimulus halves, and gains were computed in a four-dimensional response basis.

Table S12: **Direct response-space coverage on TVSD.** Per-stimulus halves, $d = 4$; parentheses report gain relative to adding held-out real responses.

Metric	Method or reference	Real-response Δ	Matched-state Δ	Cross-state Δ
Log det	Real responses	$+0.5058 \pm 0.0360$	–	–
	BrainStateDiff	–	$\mathbf{+0.4466 \pm 0.3628}$ $\mathbf{(0.88\times)}$	$\mathbf{+0.5432 \pm 0.4058}$ $\mathbf{(1.07\times)}$
	LFADS	–	$+0.8841 \pm 0.4144$ $(1.75\times)$	$+0.9044 \pm 0.4188$ $(1.79\times)$
Participation ratio	Real responses	$+0.1756 \pm 0.0022$	–	–
	BrainStateDiff	–	$+0.1231 \pm 0.0057$ $(0.70\times)$	$\mathbf{+0.1373 \pm 0.0098}$ $\mathbf{(0.78\times)}$
	LFADS	–	$\mathbf{+0.1242 \pm 0.0642}$ $\mathbf{(0.71\times)}$	$+0.1283 \pm 0.0648$ $(0.73\times)$

Table S13: **Direct response-space coverage on mouse striatum.** Parentheses report gain relative to adding held-out real responses.

Metric	Method or reference	Real-response Δ	Matched-state Δ	Cross-state Δ
Log det	Real responses	$+0.3327 \pm 0.1997$	–	–
	BrainStateDiff	–	$\mathbf{+0.2512 \pm 0.3305}$ $\mathbf{(0.76\times)}$	$\mathbf{+0.3460 \pm 0.5426}$ $\mathbf{(1.04\times)}$
	LFADS	–	-0.2581 ± 0.2686 $(-0.78\times)$	$+0.1861 \pm 0.5435$ $(0.56\times)$
Participation ratio	Real responses	$+0.0959 \pm 0.1145$	–	–
	BrainStateDiff	–	$\mathbf{+0.0493 \pm 0.0823}$ $\mathbf{(0.51\times)}$	$\mathbf{+0.0584 \pm 0.1163}$ $\mathbf{(0.61\times)}$
	LFADS	–	$+0.0355 \pm 0.0755$ $(0.37\times)$	$+0.1686 \pm 0.1576$ $(1.76\times)$

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction state the main contribution, BrainStateDiff, and describe its intended scope: state-conditioned synthetic neural data generation using cross-state and matched-state sampling. The claims are supported by the simulated dataset and three empirical datasets presented in Section 4.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: The Discussion describes limitations of BrainStateDiff, including dependence on measured state variables, assumptions behind cross-state recombination, and evaluation across a limited set of recording modalities and downstream tasks.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [N/A]

Justification: The paper does not present formal theoretical results, theorems, or proofs. The mathematical content is limited to the model formulation, training objective, and sampling definitions used in the experiments.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: The appendix describes the datasets, preprocessing, model architectures, diffusion objectives, sampling procedures, downstream decoders, train/validation/test splits, and evaluation metrics needed to reproduce the main experimental results. The main text also identifies the experimental settings and baselines that support the paper's claims.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).

(d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: Links to the code and mouse wide-field calcium imaging dataset are provided in footnote 1.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: The appendix specifies dataset splits, preprocessing, model architectures, optimizer choices, training schedules, diffusion sampling settings, augmentation protocols, and downstream evaluation details for each experiment. Hyperparameters and baseline settings are described in the corresponding dataset-specific appendix sections.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

Justification: The main experimental figures report variability using error bars, with captions specifying mean and s.e.m. where applicable. The reported variability is computed across relevant experimental units such as subjects, sessions, images, or downstream training seeds, depending on the experiment.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Appendix E reports the compute environment and GPU memory requirements needed to reproduce the experiments. The wide-field decoding analysis requires 80 GB GPU memory, while the remaining computational tasks use institutional compute clusters with GPUs of at least 24 GB memory.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics https://neurips.cc/public/EthicsGuidelines?

   Answer: [Yes]

   Justification: We confirm that there are no harms in our code and our code follows the NeurIPS Code of Ethics.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.

- If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [N/A]

    Justification: Our work is fundamental research focused on understanding state-dependent neural dynamics in systems and computational neuroscience, without direct societal impacts beyond advancing basic neuroscience knowledge.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: Our work focuses on fundamental neuroscience research methods for analyzing state-dependent neural dynamics from mice and monkeys. There are no foreseeable dual-use applications or potential for misuse that would require specific safeguards.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: We used publicly available code for baseline comparison from appropriate repositories with proper citations. Publicly available monkey electrophysiology datasets and mouse 2p striatal datasets are appropriately specified in the paper and referenced. The ViT-MAE is also properly cited.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
    - If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: The appendix provides detailed model hyperparameters, preprocessing procedures, and implementation details sufficient for reproducing our results.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: No crowdsourcing or human subjects were involved in this work. Our study uses only synthetic dataset, a public monkey electrophysiology dataset, a mouse 2p striatal dataset, and laboratory mice experiments.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.

- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: No crowdsourcing or human subjects were involved in this work. Our study uses only synthetic dataset, a public monkey electrophysiology dataset, a mouse 2p striatal dataset, and laboratory mice experiments.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [N/A]

    Justification: We did not use LLMs as part of the core methodology or experimental approach. Our work relies entirely on traditional methods without any LLM components influencing the research design, implementation, or outcomes. LLM is only used for writing, grammar check, and formatting.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.