

## Appendix

### A Model-Brain Alignment per Transition and per Module

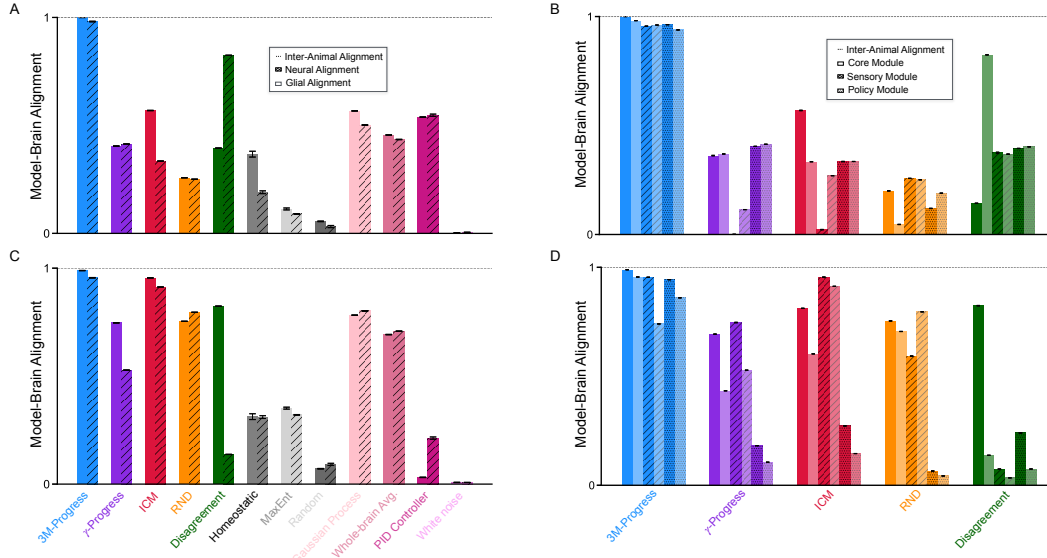


Figure 6: Model-Brain alignment for active and passive transitions and per module, excluding per module read-in and readout layers where applicable. A) Alignment for Active Transitions. B) Alignment per Agent Module for Active Transitions. C) Alignment for Passive Transitions. D) Alignment per Agent Network Module for Passive Transitions.

### B Latent Dynamics of Baseline Agents

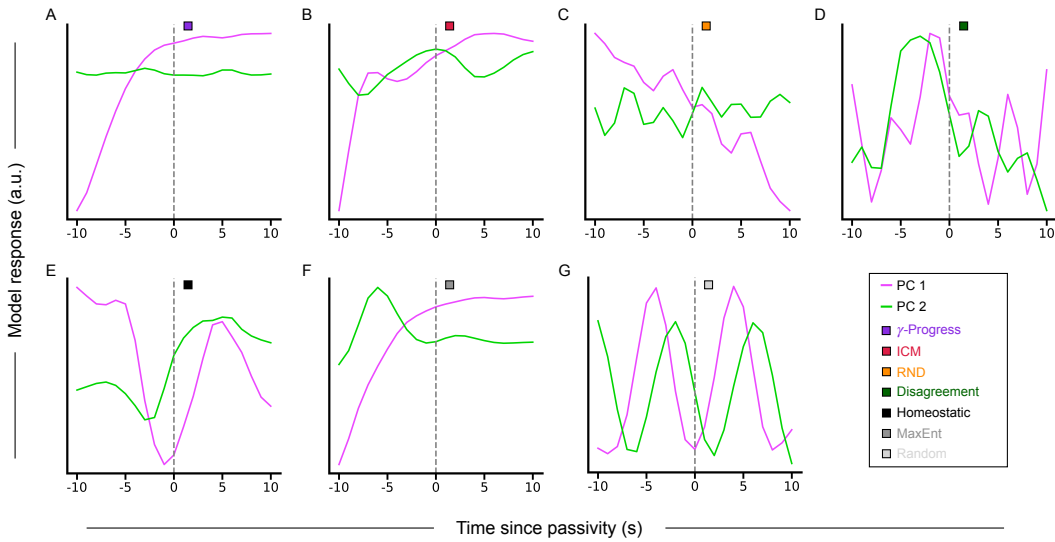


Figure 7: Latent dynamics of each agent-based control model. Dashed line indicates time since passivity. Coloring of PCs neural-glial cell-types (Fig. 2 Fig. 5) was chosen according to cell-types of the 3M-Progress agent PCs.

## C Formal Intuitions

### C.1 Curiosity-driven Exploration is Not Enough

Consider an MDP with discount  $\gamma \in (0, 1)$ , occupancy  $d_\pi(s, a) = (1-\gamma) \sum_{t \geq 0} \gamma^t \Pr_\pi(s_t=s, a_t=a)$ , and transition kernel  $\mathcal{T}(s' | s, a)$ . Let a predictive world model with parameters  $\theta$  be trained online to minimize

$$\mathcal{L}(\pi, \theta) = \mathbb{E}_{(s,a,s') \sim d_\pi \mathcal{T}} [\ell(s, a, s'; \theta)],$$

and suppose the policy receives intrinsic reward determined by this predictor:  $r^i(s, a, s'; \theta) = g(\ell(s, a, s'; \theta))$  where  $\ell \geq 0$  and  $g: \mathbb{R}_{\geq 0} \rightarrow \mathbb{R}_{\geq 0}$  is monotone increasing with  $g(0) = 0$ . The policy is updated to maximize

$$J(\pi, \theta) = \mathbb{E}_{(s,a,s') \sim d_\pi \mathcal{T}} [r^i(s, a, s'; \theta)].$$

Under either alternating or simultaneous updates that decrease  $\mathcal{L}$  in  $\theta$  and increase  $J$  in  $\pi$ , the process cannot converge to a nontrivial, uniquely defined exploratory policy. The only stationary outcome is a degenerate collapse in which  $r^i$  is constant on the visited support; otherwise the policy-model coupling remains non-stationary and induces drift. That is, any stationary point  $(\pi^*, \theta^*)$  satisfies one of the following cases:

1. *Reward collapse.* If  $\theta^*$  minimizes  $\mathcal{L}(\pi^*, \theta)$  on the support of  $d_{\pi^*} \mathcal{T}$  and the model class is realizable on that support, then  $\ell(\cdot; \theta^*) = 0$  almost surely, hence  $r^i(\cdot; \theta^*) = 0$  almost surely and  $J(\pi, \theta^*) = 0$  for all  $\pi$ . The objective is flat and does not select a unique exploratory policy.
2. *No stationary policy.* If we freeze the model at  $\theta^*$ , then  $r^i(\cdot; \theta^*)$  is a fixed reward. If residual error remains, then  $\ell(\cdot; \theta^*)$  (hence  $r^i$ ) is not almost surely constant, so there exists a measurable set  $U$  on which the intrinsic advantage  $A_{\pi^*}^{\text{int}}(s, a) > 0$  for some actions. By the policy-gradient identity,

$$\nabla J(\pi^*, \theta^*) = (1 - \gamma) \mathbb{E}_{d_{\pi^*}} [A_{\pi^*}(s, a) \nabla \log \pi^*(a | s)].$$

For some state  $s \in U$ , suppose we increase  $\pi^*(a | s)$  slightly on an action with  $A_{\pi^*}(s, a) > 0$  and decrease it on other actions to preserve  $\sum_a \pi^*(a | s) = 1$ . This choice makes the inner product  $\langle A_{\pi^*}(s, \cdot), \nabla \log \pi^*(\cdot | s) \rangle$  strictly positive on  $U$ , hence the expectation above is positive and the policy gradient is nonzero. Therefore  $\pi^*$  is not a local maximizer of the stationary objective. Any such occupancy shift then triggers predictor updates that reduce  $r^{\text{int}}$  on  $U$ , moving the high-reward region and preventing a nontrivial fixed point.

In both cases, these intrinsic signals do not converge to a stable, uniquely-defined exploratory policy. Either training drives the intrinsic signal to a constant (degenerate) value on the visited distribution, or residual heterogeneity in the reward keeps creating ascent directions that are then neutralized by predictor updates, preventing stabilization.

### C.2 3M-Progress Partitions the Behavioral Space

Consider two reward-free MDPs,  $\mathcal{M}_1 = (\mathcal{S}, \mathcal{A}, \mathcal{T}_1, p_0)$  and  $\mathcal{M}_2 = (\mathcal{S}, \mathcal{A}, \mathcal{T}_2, p_0)$ , that differ only in their transition densities. Further, suppose there exists a set  $U \subset \mathcal{S} \times \mathcal{A}$  such that  $\mathcal{T}_1(\cdot | s, a) = \mathcal{T}_2(\cdot | s, a)$  almost everywhere in  $s'$  for all  $(s, a) \in U$ . Let  $p(\cdot | s, a)$  and  $q(\cdot | s, a)$  be world models trained on data from  $\mathcal{T}_1$  and  $\mathcal{T}_2$ , respectively, with pointwise consistency:

$$\mathcal{D}_{\text{KL}}(\mathcal{T}_1(\cdot | s, a) \| p(\cdot | s, a)) \xrightarrow{P} 0, \quad \mathcal{D}_{\text{KL}}(\mathcal{T}_2(\cdot | s, a) \| q(\cdot | s, a)) \xrightarrow{P} 0.$$

Assume for each  $(s, a)$ , both kernels share the same support with no vanishing probabilities (i.e.  $\exists c > 0$  s.t.  $\mathcal{T}_2(s' | s, a) \geq c$  whenever  $\mathcal{T}_1(s' | s, a) > 0$ ). Then it follows that for all  $(s, a) \in U$ ,

$$\mathcal{D}_{\text{KL}}(p(\cdot | s, a) \| q(\cdot | s, a)) \xrightarrow{P} 0,$$

and that for all  $(s, a) \in U^c$ ,

$$\mathcal{D}_{\text{KL}}(p(\cdot | s, a) \| q(\cdot | s, a)) \xrightarrow{P} \mathcal{D}_{\text{KL}}(\mathcal{T}_1 \| \mathcal{T}_2) > 0.$$

### C.3 Beyond Futility-induced Passivity

Although we demonstrate 3M-Progress on a specific autonomous behavior known as futility-induced passivity, our algorithm applies to any exploration behavior in which a dynamics niche is reasonably specified. In our experiments, futility-induced passivity arises completely from the choice of the pretraining environment and the online learning environment; we choose these environments such that the transition dynamics between environments agree when the agent is passive, thus defining the ecological niche that guides exploration (see Appendix C.2; simply put, we choose environments such that there exists a subset  $U$  in which the transition dynamics locally agree). In general, the pretraining environment should encode a meaningful dynamics prior the agent can use for continual learning with environments whose physics vary systematically from the pretraining environment.

For example, suppose we pretrain an agent and world model on a foraging task (e.g. the virtual rodent environment in [66]). In a new environment that includes the opportunity for foraging and any of its constituent locomotor primitives (running, turning, jumping, etc.), the basic distribution-matching objective of 3M-Progress rewards the agent for trajectories whose dynamics are predictable under the pretrained world model (namely, foraging and constituent motor primitives):

$$r_t^i = f(\hat{\epsilon}_t - \epsilon_t); \quad \epsilon_t = \mathcal{D}_{\text{KL}}(\omega_{\theta'} \parallel \omega_{\theta}); \quad \hat{\epsilon}_t = (1 - \gamma)\hat{\epsilon}_t + \gamma\epsilon_t$$

where  $\theta'$  is learned online and  $\theta$  is pretrained. Niche-aware exploration is primarily mediated by the activation function  $f$ . When  $f$  is monotonic non-decreasing, such as a rectified linear unit, 3M-Progress is niche-seeking with the niche defined by the dynamics prior  $\omega_{\theta}$ . Conversely, if  $f$  is monotonic non-increasing, 3M-Progress is niche-avoiding and explores dynamics outside the prior. Non-monotonic functions allow some amount of symmetry between niche-seeking and niche-avoidance depending on the specific function shape.

The utility of this approach can also be appreciated when the pretraining stage involves a large diversity of dynamics, either from multiple environments and tasks or a single environment with multiple tasks. A world model with sufficient computational capacity that captures these diverse dynamics can be flexibly used in new environments to motivate exploration in a variety of ways. In the simplest case, for example, suppose we pretrain an ensemble of dynamics models  $\{\omega_{\theta_j}\}_{j=1}^N$  on  $N$  separate environments and tasks. Maintaining independent temporal filters  $\hat{\epsilon}_t^j$  for each prior, a deterministic intrinsic motivation can be defined as  $r_t^i = \max\{f(\hat{\epsilon}_t^j - \epsilon_t^j)\}_{j=1}^N$ . Alternatively, one can imagine various sampling schemes over the ensemble in order to drive specific exploration styles, such as  $\epsilon$ -greedy or max-entropy. This extends 3M-Progress to niche-aware exploration over multiple niches, thereby allowing the flexibility of multiple modes of behavior in a single objective function. Each addition of a dynamics prior further partitions the behavioral space in the online environment, and allows the dynamics characteristic of each pretraining environment to be composed to form an exploration landscape of attractors (niche-seeking) or repellers (niche-avoidance) for online learning.

## D Description of Control Models

### D.1 Model-based Controls

**The Intrinsic Curiosity Module (ICM)** [3] defines intrinsic reward as the Shannon surprise of the forward model,  $r_t^i := -\log \omega_{\theta}(\phi_t^I \mid \phi_t, \mathbf{a}_t)$ .  $I$  denotes an augmented inverse feature space that is learned on-top of sensory embeddings from  $\phi$ —ICM trains an additional embedding layer  $\phi^I$  using an inverse dynamics model parameterized by  $\theta_I$  by optimizing an MSE loss  $L(\theta_I) = \mathbb{E}_{\pi_{\theta}} \|f(\theta_I; \phi_t^I, \phi_{t+1}^I) - \mathbf{a}_t\|_2^2$ .

**Random Network Distillation (RND)** [4] defines a fixed random nonlinear projection of sensory features  $g(\phi)$  and trains a predictor network  $\hat{g}$  using an MSE loss  $r_t^i := L(\theta_{\text{RND}}) = \mathbb{E}_{\pi_{\theta}} \|g(\phi_t) - \hat{g}(\theta_{\text{RND}}; \phi_t)\|_2^2$ . With the distillation objective as the intrinsic reward, RND does not reinforce behaviors by scoring their predictability by a forward dynamics model as in ICM; instead, the random memory provides a simple exploration bonus for visiting novel states under the policy distribution.

**Disagreement** [5] learn an ensemble of world models  $\{\omega_{\theta_j}\}_{j=1}^N$  and defines intrinsic reward as  $r_i^t := \text{Var}(\{\mu_{\theta_j} : j \in [N]\})$  for  $N$  randomly initialized world models. Ensemble variances scores the stochasticity of the environment and reinforces state-action pairs for which models disagree.

**$\gamma$ -Progress** [7] leverages the temporal history of Shannon surprise to define intrinsic reward using prediction gain,  $r_i^t := \log \frac{\omega_{\theta_{new}}}{\omega_{\theta_{old}}}$ , where  $\theta_{new}$  parameterized a world model after learning on new transitions withheld from an lagging model  $\theta_{old}$ .

## D.2 Model-Free Controls

**Homeostatic Agent** One straightforward way to achieve a passive behavioral transition is to simply add an action-cost that outweighs any other positive reward signal. In the presence of a fixed or nonexistent extrinsic reward signal which the agent has no control over (such as in the open-loop protocol), an action-cost encourages the agent to become passive, corresponding to a metabolic constraint or homeostatic regulation of energy. We implement this cost as the magnitude of the force exerted by the agent’s motors,  $c(a_t) = \lambda \|\mathbf{a}_t\|_2$ . In all our experiments, we set  $\lambda = 1$ .

**Maximum Entropy Agent** Maximum entropy RL is a general exploration strategy that provides a bonus reward proportional to the entropy of the current policy. That is,  $r_i^t = \lambda \mathcal{H}[\pi(\mathbf{a}_t | \mathbf{s}_t)]$ . In all our experiments, we set  $\lambda = 1$ .

**Random Agent** To test a random baseline model for embodied control, we use a randomly initialized agent with the same model architecture (described in section 3 Figure 2A, and in PPO implementation details below).

**Whole-brain Average** We use the average recorded neural-glial activity during a specified behavioral transition in one larval zebrafish to predict whole-brain neural-glial activity recorded from another larval zebrafish undergoing the same transition. The alignment under the metric described in section 3 is computed using the average response of each cell-type (neural and glial) from the source animal to predict the corresponding cell-type in the target animal. We use two subjects and report the total alignment as the sample-weighted average between scores from both source-target pairs.

**Gaussian-Process** We fit a separate Gaussian Process (GP) to each cell-type (neural and glial) for each subject, using a radial basis function (RBF) kernel and centering the prior mean at the average whole-brain response. Alignment under the metric described in section 3 is computed between the whole-brain data from the target cell-type from an individual subject and its corresponding GP as the source. We use two subjects and report the total alignment as the sample-weighted average between scores from both source-target pairs.

**White-noise** At each timestep  $t$ , the model’s predicted next state is give by  $x_{t+1} = x_t + \eta_t$ , where  $\eta_t \sim \mathcal{N}(0, 1)$  is white-noise. Because the metric that saturates inter-animal alignment is correlation-based, the mean and variance of this random walk are arbitrary.

**PID Controller** To implement the circuit-based word model proposed by Mu et al. [2] for the zebrafish brain’s transition from active to passive states, we employ a threshold-based (“bang–bang”) controller that switches the fish’s swim power  $P(t)$  between an active waveform  $P_{base}(t)$  and complete cessation ( $P(t) = 0$ ) once a cumulative “futility” signal exceeds a fixed GABAergic threshold. This controller can be interpreted as the high-gain limit of a saturated PID controller.

Perceived hydrostatic drift is modeled as a constant  $v_d$ . The motor plant converts swim power into a counter-drift locomotor velocity with gain  $g_{MS}$ ,

$$v_s(t) = v_d - g_{MS}P(t).$$

Visual mismatch is the product of forward stimulus velocity and ongoing motor drive but is rectified to ignore overshoot,

$$e(t) = \begin{cases} v_s(t)P(t), & \text{if } v_s(t) > 0, \\ 0, & \text{otherwise.} \end{cases}$$

A leaky integrator with time constant  $\tau_F$  accumulates the mismatch (i.e., the futility):

$$\dot{F}(t) = -\lambda_F F(t) + k_F e(t),$$

where  $\lambda_F = 1/\tau_F$  and  $k_F$  is a gain. A second leaky integrator converts sustained futility into an inhibitory drive,

$$\dot{G}(t) = -\lambda_G G(t) + k_G \max(0, F(t) - \theta_F),$$

$$P(t) = \begin{cases} P_{\text{base}}(t), & G(t) \leq \theta_G, \\ 0, & G(t) > \theta_G. \end{cases}$$

To mimic experimental perturbations we set  $g_{\text{MS}} = 0$  between  $t_{\text{OL on}}$  and  $t_{\text{OL off}}$ , effectively clamping optic-flow feedback.

## E Implementation Details

All code can be found in [https://github.com/neuroagents-lab/autonomous\\_zebrafish](https://github.com/neuroagents-lab/autonomous_zebrafish)

**Proximal Policy Optimization (PPO)** In all our experiments, we train our agents with PPO using a clipped surrogate objective [55] with  $\epsilon^{CLIP} = 0.2$  and normalized advantage function computed using Generalized Advantage Estimation (GAE) [67] with  $\lambda^{GAE} = 0.95$ . The policy network is an MLP with two hidden layers [128, 64] optimized by Adam [68] with learning rate  $\alpha = 0.0003$  and gradients computed over 5 epochs of 1000-step trajectories with a 250-step batch size vectorized across 64 environments. This MLP parameterizes the policy as a 5-dimensional diagonal Gaussian distribution which is sampled to produce actions during training. For evaluation (the experiments in section 5), the policy is deterministic by taking actions as the mean of the distribution. Actions take the form of continuous real-valued torques on  $[-1, 1]$ . The intrinsic rewards are normalized by dividing by a running estimate of the standard deviation of the sum of discounted rewards with discount factor  $\gamma = 0.99$ , which then supervises the value network MLP with two hidden layers [128, 64] to estimate the expected discounted return using this same discount factor. Both policy and value networks compute on hidden states from separate LSTM blocks (Figure 2A) using a shared embedding from the sensory feature extractors. Image embeddings are obtained from 64x64 pixel observations passed through a three-layer ResNet resulting in outputs with a spatial resolution of 16x16 (closely matching the visual acuity of larval zebrafish at the final layer). Image embeddings were flattened and concatenated with proprioceptive features including joint positions and rotational velocities and their embeddings from a two-layer MLP with hidden size [64, 64].

**Intrinsic Drive Module (IDM)** The IDM architecture details vary depending on which intrinsic drive it implements, and is described on a case-by-case basis in the sections below. Here, we describe the commonalities between intrinsic drives, which include the optimizer, forward dynamics loss, general forward model architecture, and memory buffer parameters. Each forward dynamics model across intrinsic drives is implemented as a two-layer MLP with hidden sizes [512, 512], trained to predict the true observation one time-step into the future from the current observation using an MSE loss (where inference is done in feature-space), and optimized using Adam [68] with learning rate  $\alpha = 0.001$ . The IDM maintains a memory buffer of the last 100 observation embeddings from each environment in the vector and trains its constituent networks on this buffer every 20 steps. The IDM is trained for  $1e^5$  steps before the intrinsic rewards are observed by the agent.

**3M-Progress** The ethological memory is created (with the default configuration outlined above) by training a swimmer agent on a simple navigation task in a head-free version of the experimental protocol outlined in section 3. This environment is chosen to provide the forward model with a similar visual input space as the head-fixed version while maintaining the ethology of unconstrained swimming in which the agent experiences naturalistic fluid forces and positional displacement in response to swim commands. The task is implemented as shaped reward proportional to the distance of the agent from a target location that is in front of the agent, such that the swim-to-target behavior results in stabilizing the constant backwards flow of the high-contrast grating. The episode is long

enough that optimizing this reward allows the agent to become passive was the target is reached. Together, our setup allows the agent to experience state-action-state triplets (current state, current action, and resulting state) associated with active and passive behaviors. This provides a close correspondence between sensory-motor coupling in our virtual ethological environment and the closed-loop experimental condition in Mu et al. [2], where larval zebrafish swim against the passive backwards flow of high-contrast gratings motivated by positional homeostasis. Although our agent is motivated to swim by a different signal than its biological twin (i.e., moving towards a target location rather than a homeostatic drive that resists displacement from an opposing current), the design of the virtual environment renders the sensory-motor stream experienced by the internal world model equivalent between scenarios, since in both cases optic flow is counteracted by forward swim motion. Because this sensory-motor stream and its resulting world-model alone define the ethological memory, the discrepancy in the signal that drove behavior has no bearing on training the new policy and value network in the open-loop environment (Figures 1D, 2B).

In the open-loop environment, the agent randomly initializes a new world-model memory that is trained online with the default configuration outlined in the IDM section. The ethological world-model memory is loaded from earliest checkpoint where the agent achieved optimal swim-to-target behavior and its weights are frozen. The model-memory-mismatch is computed as the MSE between the predictions from each memory, filtered by an exponential moving average with timescale  $\gamma = 0.99$ , and the difference filtered and unfiltered predictions are passed through an  $L_1$  activation.

**Random Network Distillation (RND)** [4] For RND both target and predictor networks are implemented using the default configuration in the IDM section. The predictor network is trained to predict random feature projections from the target as described in section 3.

**Intrinsic Curiosity Module (ICM)** [3] In addition to a forward model implemented using the default configuration in the IDM section, the ICM implements an inverse dynamics model as an MLP with an identical architecture and optimization routine to train a one-layer MLP on top of sensory features. The forward and inverse networks are cotrained by minimizing a joint objective  $\beta L_F + (1 - \beta)L_I$ , where  $L_F$  and  $L_I$  are the forward and inverse MSE loss functions, respectively. In our experiments, we set  $\beta = 0.2$ .

**Disagreement** [5] We use  $N = 3$  randomly initialized independent forward models using the default configuration described in the IDM section. Disagreement is computed as the mean variance across feature dimensions.

**$\gamma$ -Progress** [6] Using a randomly initialized forward model implemented using the default configuration in the IDM section, the trailing memory is created by copying these initial weights and updating them using an exponential moving average with timescale  $\gamma$ . In all our experiments, we use  $\gamma = 0.99$ .

## F Inter-Subject Noise Correction Derivation

Herein we describe how the metric  $\mathcal{M}$  should correct for noise if there is trial-to-trial variability. This is unified and adapted from Nayebi\* et al. [62], Nayebi et al. [63, 69]. If you prefer to skip the derivation, for common choices of metric  $\mathcal{M}$ , such as Pearson correlation, RSA, and especially any metric that satisfies transitive closure [70], one will need to correct by the square root of the product of the mapping consistency and internal consistency of the units, in order to properly approximate the true value of  $\mathcal{M}$  in the limit of infinite trials.

To make this correction explicit, suppose we have neural responses from two animals (or subjects) A and B. Let  $t_i^p$  be the vector of true responses (either at a given time bin or averaged across a set of time bins) of animal  $p \in \mathcal{A} = \{A, B, \dots\}$  on stimulus set  $i \in \{\text{train}, \text{test}\}$ . Of course, we only receive noisy observations of  $t_i^p$ , so let  $s_{j,i}^p$  be the  $j$ th set of  $n$  trials of  $t_i^p$ . Finally, let  $M(x; y)_i$  be the predictions of a mapping  $M$  (e.g., PLS, or any type of regression) when trained on input  $x$  to match output  $y$  and tested on stimulus set  $i$ . For example,  $M(t_{\text{train}}^A; t_{\text{train}}^B)_{\text{test}}$  is the prediction of mapping  $M$  on the test set stimuli trained to match the true neural responses of animal B given, as input, the true neural responses of animal A on the train set stimuli. Similarly,  $M(s_{1,\text{train}}^A; s_{1,\text{train}}^B)_{\text{test}}$  is the prediction of mapping  $M$  on the test set stimuli trained to match the trial-average of noisy sample 1

on the train set stimuli of animal B given, as input, the trial-average of noisy sample 1 on the train set stimuli of animal A. Then we have that:

$$\begin{aligned} \mathcal{M}_{\text{true}} &:= \langle \text{Corr}(M(t_{\text{train}}^A; t_{\text{train}}^B)_{\text{test}}, t_{\text{test}}^B) \rangle \\ \sim \widehat{\mathcal{M}}_{\text{est}} &:= \left\langle \frac{\overbrace{\text{Corr}(M(s_{1,\text{train}}^A; s_{1,\text{train}}^B)_{\text{test}}, s_{2,\text{test}}^B)}^{\text{predictivity}}}{\underbrace{\widetilde{\text{Corr}}(M(s_{1,\text{train}}^A; s_{1,\text{train}}^B)_{\text{test}}, M(s_{2,\text{train}}^A; s_{2,\text{train}}^B)_{\text{test}})}_{\text{mapping consistency}} \times \underbrace{\widetilde{\text{Corr}}(s_{1,\text{test}}^B, s_{2,\text{test}}^B)}_{\text{internal consistency}}} \right\rangle, \end{aligned} \quad (4)$$

where the average  $\langle \cdot \rangle$  is taken over bootstrapped split-half trials and train-test splits, and  $\text{Corr}(\cdot, \cdot)$  denotes the Pearson correlation of the two quantities.  $\widetilde{\text{Corr}}(\cdot, \cdot)$  denotes the Spearman-Brown corrected value of the original quantity (since it is computed on split-halves of the trials, unlike the numerator, which is evaluated on the full trial set). The analogous correction for RSA holds, where the RDM/RSM of the responses is instead used for  $s$ , and  $M$  is the identity mapping,  $M(x; \cdot)_{\text{test}} \equiv x_{\text{test}}$ . When constructing  $\widehat{\mathcal{M}}_{\text{est}}$  for model-brain mappings (rather than brain-brain mappings), we just replace A with the model responses, which are deterministic.

The above correction in (4) is fully implemented in the `brainmodel_utils` package ([https://github.com/neuroagents-lab/brainmodel\\_utils](https://github.com/neuroagents-lab/brainmodel_utils)), specifically in the `get_linregress_consistency` function. This function can be imported as follows:

```
from brainmodel_utils.metrics.consistency import get_linregress_consistency
```

The `r_xy_n_sb` value returned by this function corresponds to the ratio in (4). Refer to the [README](#) and the function docstring for usage details across a range of linearly regressed and non-regressed (e.g. RSA) metrics.

## E.1 Single Subject Pair

Suppose we have neural responses from two animals (or subjects) A and B. Let  $t_i^p$  be the vector of true responses (either at a given time bin or averaged across a set of time bins) of animal  $p \in \mathcal{A} = \{A, B, \dots\}$  on stimulus set  $i \in \{\text{train}, \text{test}\}$ . Of course, we only receive noisy observations of  $t_i^p$ , so let  $s_{j,i}^p$  be the  $j$ th set of  $n$  trials of  $t_i^p$ . Finally, let  $M(x; y)_i$  be the predictions of a mapping  $M$  (e.g., PLS) when trained on input  $x$  to match output  $y$  and tested on stimulus set  $i$ . For example,  $M(t_{\text{train}}^A; t_{\text{train}}^B)_{\text{test}}$  is the prediction of mapping  $M$  on the test set stimuli trained to match the true neural responses of animal B given, as input, the true neural responses of animal A on the train set stimuli. Similarly,  $M(s_{1,\text{train}}^A; s_{1,\text{train}}^B)_{\text{test}}$  is the prediction of mapping  $M$  on the test set stimuli trained to match the trial-average of noisy sample 1 on the train set stimuli of animal B given, as input, the trial-average of noisy sample 1 on the train set stimuli of animal A.

With these definitions in hand, the inter-animal mapping consistency from animal A to animal B corresponds to the following “true” quantity to be estimated by  $\widehat{\mathcal{M}}_{\text{est}}$  in the limit of infinite trials:

$$\mathcal{M}_{\text{true}} := \text{Corr}(M(t_{\text{train}}^A; t_{\text{train}}^B)_{\text{test}}, t_{\text{test}}^B), \quad (5)$$

where  $\text{Corr}(\cdot, \cdot)$  is the Pearson correlation across a stimulus set. In what follows, we will argue that Eq (5) can be approximated with the following ratio of measurable quantities, where we split in half and average the noisy trial observations, indexed by 1 and by 2:

$$\begin{aligned} \mathcal{M}_{\text{true}} &:= \text{Corr}(M(t_{\text{train}}^A; t_{\text{train}}^B)_{\text{test}}, t_{\text{test}}^B) \\ \sim \widehat{\mathcal{M}}_{\text{est}} &:= \frac{\text{Corr}(M(s_{1,\text{train}}^A; s_{1,\text{train}}^B)_{\text{test}}, s_{2,\text{test}}^B)}{\sqrt{\text{Corr}(M(s_{1,\text{train}}^A; s_{1,\text{train}}^B)_{\text{test}}, M(s_{2,\text{train}}^A; s_{2,\text{train}}^B)_{\text{test}})} \times \text{Corr}(s_{1,\text{test}}^B, s_{2,\text{test}}^B)}. \end{aligned} \quad (6)$$

In words, the inter-animal consistency (i.e., the quantity on the left side of Eq (6)) corresponds to the predictivity of the mapping on the test set stimuli from animal A to animal B on two different (averaged) halves of noisy trials (i.e., the numerator on the right side of Eq (6)), corrected by the



square root of the mapping reliability on animal A's responses to the test set stimuli on two different halves of noisy trials multiplied by the internal consistency of animal B.

We justify the approximation in Eq (6) by gradually replacing the true quantities (t) by their measurable estimates (s), starting from the original quantity in Eq (5). First, we make the approximation that:

$$\text{Corr} (M (t_{\text{train}}^A; t_{\text{train}}^B)_{\text{test}}, s_{2,\text{test}}^B) \sim \text{Corr} (M (t_{\text{train}}^A; t_{\text{train}}^B)_{\text{test}}, t_{\text{test}}^B) \times \text{Corr} (t_{\text{test}}^B, s_{2,\text{test}}^B), \quad (7)$$

by the transitivity of very positive correlations. Namely, in scenarios where correlations are very close to 1, a form of transitivity holds, meaning if variable  $A$  is highly correlated with variable  $B$ , and variable  $B$  with variable  $C$ , then variable  $A$  is also highly correlated with variable  $C$ . This is the desired situation, as low or negative correlations indicate neurons that are not self-consistent. Moreover, calculating certain metrics in these cases can result in undefined values due to operations like taking the square root of a negative number. Assuming high correlations is reasonable, especially when the number of stimuli is large. Next, by transitivity and normality assumptions in the structure of the noisy estimates and since the number of trials ( $n$ ) between the two sets is the same, we have that:

$$\begin{aligned} \text{Corr} (s_{1,\text{test}}^B, s_{2,\text{test}}^B) &\sim \text{Corr} (s_{1,\text{test}}^B, t_{\text{test}}^B) \times \text{Corr} (t_{\text{test}}^B, s_{2,\text{test}}^B) \\ &\sim \text{Corr} (t_{\text{test}}^B, s_{2,\text{test}}^B)^2. \end{aligned} \quad (8)$$

In words, Eq (8) states that the correlation between the average of two sets of noisy observations of  $n$  trials each is approximately the square of the correlation between the true value and average of one set of  $n$  noisy trials. Therefore, combining Eq (7) and Eq (8), it follows that:

$$\text{Corr} (M (t_{\text{train}}^A; t_{\text{train}}^B)_{\text{test}}, t_{\text{test}}^B) \sim \frac{\text{Corr} (M (t_{\text{train}}^A; t_{\text{train}}^B)_{\text{test}}, s_{2,\text{test}}^B)}{\sqrt{\text{Corr} (s_{1,\text{test}}^B, s_{2,\text{test}}^B)}}. \quad (9)$$

From the right side of Eq (9), we can see that we have removed  $t_{\text{test}}^B$ , but we still need to remove the  $M (t_{\text{train}}^A; t_{\text{train}}^B)_{\text{test}}$  term, as this term still contains unmeasurable (i.e., true) quantities. We apply the same two steps, described above, by analogy, though these approximations may not always be true (they are, however, true for Gaussian noise):

$$\begin{aligned} \text{Corr} (M (s_{1,\text{train}}^A; s_{1,\text{train}}^B)_{\text{test}}, s_{2,\text{test}}^B) &\sim \text{Corr} (s_{2,\text{test}}^B, M (t_{\text{train}}^A; t_{\text{train}}^B)_{\text{test}}) \\ &\quad \times \text{Corr} (M (t_{\text{train}}^A; t_{\text{train}}^B)_{\text{test}}, M (s_{1,\text{train}}^A; s_{1,\text{train}}^B)_{\text{test}}) \\ &\sim \text{Corr} (M (s_{1,\text{train}}^A; s_{1,\text{train}}^B)_{\text{test}}, M (s_{2,\text{train}}^A; s_{2,\text{train}}^B)_{\text{test}}) \\ &\sim \text{Corr} (M (s_{1,\text{train}}^A; s_{1,\text{train}}^B)_{\text{test}}, M (t_{\text{train}}^A; t_{\text{train}}^B)_{\text{test}})^2, \end{aligned}$$

which taken together implies the following:

$$\text{Corr} (M (t_{\text{train}}^A; t_{\text{train}}^B)_{\text{test}}, s_{2,\text{test}}^B) \sim \frac{\text{Corr} (M (s_{1,\text{train}}^A; s_{1,\text{train}}^B)_{\text{test}}, s_{2,\text{test}}^B)}{\sqrt{\text{Corr} (M (s_{1,\text{train}}^A; s_{1,\text{train}}^B)_{\text{test}}, M (s_{2,\text{train}}^A; s_{2,\text{train}}^B)_{\text{test}})}}. \quad (10)$$

Eq (9) and Eq (10) together imply the final estimated quantity given in Eq (6).

## F.2 Multiple Subject Pairs

For multiple animals, we consider the average of the true quantity for each target in B in Eq (5) across source animals A in the ordered pair (A, B) of animals A and B:

$$\begin{aligned} \mathcal{M}_{\text{true}} &:= \langle \text{Corr} (M (t_{\text{train}}^A; t_{\text{train}}^B)_{\text{test}}, t_{\text{test}}^B) \rangle_{A \in \mathcal{A}: (A,B) \in \mathcal{A} \times \mathcal{A}} \\ &\sim \widehat{\mathcal{M}}_{\text{est}} := \left\langle \frac{\text{Corr} (M (s_{1,\text{train}}^A; s_{1,\text{train}}^B)_{\text{test}}, s_{2,\text{test}}^B)}{\sqrt{\widetilde{\text{Corr}} (M (s_{1,\text{train}}^A; s_{1,\text{train}}^B)_{\text{test}}, M (s_{2,\text{train}}^A; s_{2,\text{train}}^B)_{\text{test}})} \times \widetilde{\text{Corr}} (s_{1,\text{test}}^B, s_{2,\text{test}}^B)} \right\rangle_{A \in \mathcal{A}: (A,B) \in \mathcal{A} \times \mathcal{A}}. \end{aligned}$$



We also bootstrap across trials, and have multiple train/test splits, in which case the average on the right hand side of the equation includes averages across these as well.

Note that each neuron in our analysis will have this single average value associated with it when *it* was a target animal (B), averaged over source animals/subsampled source neurons, bootstrapped trials, and train/test splits. This yields a vector of these average values, which we can take median and standard error of the mean (s.e.m.) over, as we do with standard explained variance metrics.

### F.3 RSA

We can extend the above derivations to other commonly used metrics for comparing representations that involve correlation. Since  $\text{RSA}(x, y) := \text{Corr}(\text{RDM}(x), \text{RDM}(y))$ , then the corresponding quantity in Eq (6) analogously (by transitivity of maximally positive correlations) becomes:

$$\begin{aligned} \mathcal{M}_{\text{true}} &:= \langle \text{RSA} \left( M \left( t_{\text{train}}^A; t_{\text{train}}^B \right)_{\text{test}}, t_{\text{test}}^B \right) \rangle_{A \in \mathcal{A}: (A, B) \in \mathcal{A} \times \mathcal{A}} \\ \sim \widehat{\mathcal{M}}_{\text{est}} &:= \left\langle \frac{\text{RSA} \left( M \left( s_{1, \text{train}}^A; s_{1, \text{train}}^B \right)_{\text{test}}, s_{2, \text{test}}^B \right)}{\sqrt{\widetilde{\text{RSA}} \left( M \left( s_{1, \text{train}}^A; s_{1, \text{train}}^B \right)_{\text{test}}, M \left( s_{2, \text{train}}^A; s_{2, \text{train}}^B \right)_{\text{test}} \right) \times \widetilde{\text{RSA}} \left( s_{1, \text{test}}^B, s_{2, \text{test}}^B \right)}} \right\rangle_{A \in \mathcal{A}: (A, B) \in \mathcal{A} \times \mathcal{A}} \end{aligned} \quad (11)$$

Note that in this case, each *animal* (rather than neuron) in our analysis will have this single average value associated with it when *it* was a target animal (B) (since RSA is computed over images and neurons), where the average is over source animals/subsampled source neurons, bootstrapped trials, and train/test splits. This yields a vector of these average values, which we can take median and s.e.m. over, across animals  $B \in \mathcal{A}$ .

For RSA, we can use the identity mapping (since RSA is computed over neurons as well, the number of neurons between source and target animal can be different to compare them with the identity mapping). As parameters are not fit, we can choose train = test, so that Eq (11) becomes:

$$\mathcal{M}_{\text{true}} := \langle \text{RSA} \left( t^A, t^B \right) \rangle_{A \in \mathcal{A}: (A, B) \in \mathcal{A} \times \mathcal{A}} \sim \widehat{\mathcal{M}}_{\text{est}} := \left\langle \frac{\text{RSA} \left( s_1^A, s_2^B \right)}{\sqrt{\widetilde{\text{RSA}} \left( s_1^A, s_2^A \right) \times \widetilde{\text{RSA}} \left( s_1^B, s_2^B \right)}} \right\rangle_{A \in \mathcal{A}: (A, B) \in \mathcal{A} \times \mathcal{A}} \quad (12)$$

### F.4 Pooled Source Animal

Often times, we may not have enough neurons per animal to ensure that the estimated inter-animal consistency in our data closely matches the “true” inter-animal consistency. In order to address this issue, we holdout one animal at a time and compare it to the pseudo-population aggregated across units from the remaining animals, as opposed to computing the consistencies in a pairwise fashion. Thus, B is still the target heldout animal as in the pairwise case, but now the average over A is over a sole “pooled” source animal constructed from the pseudo-population of the remaining animals.

Pooling data across subjects to create larger pseudopopulations is a common practice [? ], and helps researchers better isolate core representational principles that are conserved across individuals when data collection modalities limit the number of collected neurons per session.

### F.5 Spearman-Brown Correction

The Spearman-Brown correction can be applied to each of the terms in the denominator individually, as they are each correlations of observations from half the trials of the *same* underlying process to itself (unlike the numerator). Namely,

$$\widetilde{\text{Corr}}(X, Y) := \frac{2 \text{Corr}(X, Y)}{1 + \text{Corr}(X, Y)}.$$

Analogously, since  $\text{RSA}(X, Y) := \text{Corr}(\text{RDM}(x), \text{RDM}(y))$ , then we define

$$\begin{aligned}\widetilde{\text{RSA}}(X, Y) &:= \widetilde{\text{Corr}}(\text{RDM}(x), \text{RDM}(y)) \\ &= \frac{2 \text{RSA}(X, Y)}{1 + \text{RSA}(X, Y)}.\end{aligned}$$