Neural population tuning statistics as priors for multitask generalization

Zach Cohen[1,2], **Jan Drugowitsch**[1,2]
[1]Department of Neurobiology, Harvard Medical School
[2]Kempner Institute for the Study of Natural and Artifical Intelligence, Harvard University
zcohen1@g.harvard.edu, jan_drugowitsch@hms.harvard.edu

Abstract

Neural populations exhibit heterogeneous tuning: Neurons differ in their selectivity, gain, and in how their preferred inputs are distributed across stimulus or latent-variable space. While this heterogeneity has been studied extensively in the context of encoding precision, its implications for learning and generalization are less well-understood. Here, we develop a probabilistic theory that links a neural population's tuning statistics to its inductive bias: in large populations, the tuning statistics induce a kernel and, equivalently, a prior of a generative model over functions. This prior in turn predicts the population's ability to generalize across a distribution of related tasks. Our theory determines the population tuning statistics that minimize the generalization error over a distribution of functions. We further present a normative account of tuning adaptation inspired by our probabilistic approach, showing that tuning changes that maximize the marginal likelihood of observed data systematically reduce generalization error relative to non-adapting populations. Applied to the adaptation of hippocampal spatial representations in a reward learning task, we found that our theory could explain the over-representation of rewarding locations through experience in an environment with a localized reward. We further demonstrate how different components of the population's generative model differentially impact the speed and magnitude of tuning adaptation. Overall, our work proposes a novel connection between tuning variability and a population's ability to learn and to generalize across tasks and develops a Bayesian theory of tuning adaptation in neural populations during learning.

1 Introduction

Neural populations exhibit heterogeneous tuning: Neurons differ in what they are selective for and in how broadly they respond across stimulus or latent-variable space [1–10]. In CA1, for example, hippocampal "place cells" vary widely in the spatial extent of their tuning—some fire only in small spatial regions a few centimeters in size, while others are active over larger portions of the environment [6, 8–11]. Populations also exhibit heterogeneity in their tuning density: Some populations exhibit emergent, non-homogeneous tuning center distributions, over-representing behaviorally relevant regions, such as rewarding areas in the environment [12]. Intriguingly, the statistics of tuning variability in some populations is stereotyped across animals within species [8–10, 13].

Existing theories have connected tuning heterogeneity to encoding precision. Diverse tuning widths and shapes can improve representational precision [1, 10, 13, 14] and thereby reduce downstream estimation error. Such efficient coding accounts [15–20] further predict that populations adapt tuning when neural resources are limited and inputs are non-uniform. Allocating narrower or denser tuning to high-probability or low-value stimuli increases information captured per neuron [21–23]. These perspectives motivate heterogeneity and the tuning adaptation that achieves it as mechanisms for precise representation of sensory stimuli. However, such theories do not address the role of tuning

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

heterogeneity in learning, nor do they account for tuning adaptation in neural populations far from peripheral sensory areas that subserve learning flexible input-output mappings [12, 24].

Recent work has proposed a link between a population's similarity structure—how similar activity patterns are between two inputs x and x'—and the solutions found by neural populations when learning arbitrary functions from limited data [25, 26]. In particular, the population similarity structure, which is encoded in the population's kernel, functions as an inductive bias and predicts that a population with a particular kernel can generalize to some supervised tasks more efficiently than others [26]. What remains unclear, however, is how variability in a population's tuning shapes a population's kernel. Further, while kernel theories predict the generalization error of a population on a given task, it is not known how tuning should change when the inductive bias of a population is mismatched to the task the organism actually faces.

In this work, we propose a probabilistic model that clarifies the role of tuning diversity and adaptation in learning. In Sec. 2 we show how population tuning statistics and associated population geometry shape the kernel and, equivalently, induce a probabilistic prior over target functions that the population expects to encounter. This framing generalizes kernel-based theories from single-task generalization to generalization across a distribution of tasks. Indeed, Sec. 3 demonstrates that the generalization error is minimal if the prior induced by the population tuning statistics matches the task distribution. Further, our probabilistic perspective yields a normative framework for how tuning should adapt to reduce error when a population's induced prior is poorly suited for a distribution of tasks. In particular, we show that tuning adaptation driven by maximization of the marginal likelihood of observations allows a population to mitigate additional error on account of model mismatch. We demonstrate this in Sec. 4, where we apply this framework to explain hippocampal tuning adaptation during reward learning. Altogether, this work presents a novel Bayesian theory explaining the functional roles of tuning heterogeneity and adaptation for learning and generalization.

2 Population tuning statistics as priors over target functions

2.1 Setup

We assume a single-layer neural network that aims to learn a target function $y^*(\cdot)$ which maps inputs $\boldsymbol{x}$ to scalar outputs y and is drawn from some distribution $y^* \sim p(y^*)$. The network's input layer consists of a population of N neurons with population activity vector $\boldsymbol{r}(\boldsymbol{x}; \boldsymbol{\Omega})$, where $\boldsymbol{\Omega} = \{\Omega_n\}_{n=1}^N$ are the tuning parameters that determine each neuron n's mapping from input $\boldsymbol{x}$ to its activity $r(\boldsymbol{x}; \Omega_n)$. The network approximates the target function through a linear readout of the input layer, $\hat{y}(\boldsymbol{x}) = \frac{1}{\sqrt{N}} \boldsymbol{w}^\top \boldsymbol{r}(\boldsymbol{x}; \boldsymbol{\Omega})$. (Fig. 1a). Given some dataset $\mathcal{D} = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^P$, the network adjusts its readout weights $\boldsymbol{w}$ to minimize the regularized, mean-squared deviation between the true targets and the network's output on the dataset, $\varepsilon_T = \frac{1}{2P} \sum_{i=1}^P (y_i - \hat{y}(\boldsymbol{x}_i))^2 + \frac{\gamma}{2} ||\boldsymbol{w}||^2 = \frac{1}{2P} \sum_{i=1}^P (y_i - \frac{1}{\sqrt{N}} \boldsymbol{w}^\top \boldsymbol{r}(\boldsymbol{x}_i; \boldsymbol{\Omega})))^2 + \frac{\gamma}{2} ||\boldsymbol{w}||^2$, where γ controls the regularization strength (see App. C for details). We assess how well the network with learned weights $\boldsymbol{w}^*$ matches the target function through the generalization error, which is the average squared deviation between the network's predictions and the true outputs across all inputs, $\varepsilon_g(\mathcal{D}, y^*) = \left\langle (\hat{y}(\boldsymbol{x}) - y^*(\boldsymbol{x}))^2 \right\rangle_{p(\boldsymbol{x})} = \langle (\frac{1}{\sqrt{N}} \boldsymbol{w}^{*\top} \boldsymbol{r}(\boldsymbol{x}; \boldsymbol{\Omega}) - y^*(\boldsymbol{x}))^2 \rangle_{p(\boldsymbol{x})}$.

We are interested in how this generalization error depends on the tuning of individual neurons within the input population. To study this question, we assume that the function $r(\boldsymbol{x}; \Omega_n)$ that determines the response of neuron n to input $\boldsymbol{x}$ is deterministic, bounded, and parametrized by Ω_n. For each neuron, these tuning parameters $\Omega_n = (a_n, \boldsymbol{\mu}_n, \sigma_n)$ determine the neuron's peak activity, its center, and its sensitivity to change in the input, respectively. For example, the neurons could feature localized, bell-shaped tuning with $r(\boldsymbol{x}; \Omega_n) = \sqrt{a_n} \exp(-\|\boldsymbol{x} - \boldsymbol{\mu}_n\|^2 / 2\sigma_n^2)$). As an alternative, neurons could feature sigmoidal, ramping tuning curves (Fig. S1). We assume that the parameters are for each neuron drawn stochastically from a distribution $\Omega_n \sim p(\Omega)$ that is shared across all neurons in the population. Thus, while all neurons share the same tuning function $r(\cdot; \cdot)$, their tuning varies across neurons through stochastic draws of their associated tuning parameters (Fig 1b).

In this setup the generalization error $\varepsilon_g(\mathcal{D}, y^*)$ is dependent on both $\mathcal{D}$ and y^*, as well as implicitly on $p(\Omega)$ through $\boldsymbol{\Omega}$. As such, we want to study the generalization error across datasets of some size

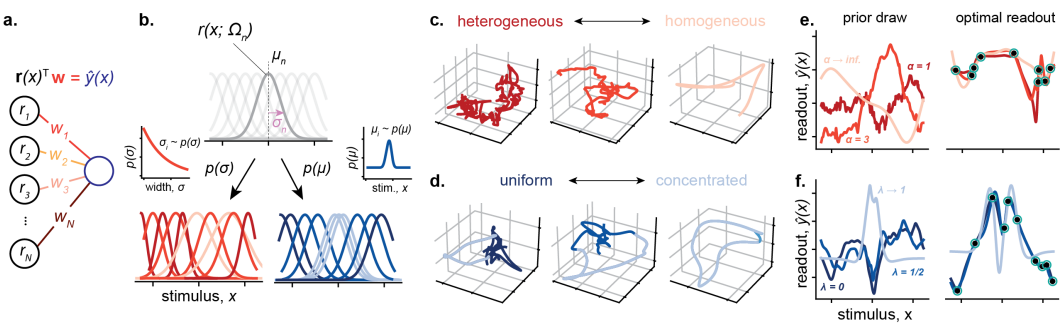

Figure 1: **Tuning statistics shape population geometry, priors over outputs, and learned outputs**. **a.** Schematic of the learning setup. **b.** Illustration of how tuning statistics shape population structure (for illustration, we do not vary tuning gains across the population). **c.** Random projections of population activity for varying degrees of bandwidth heterogeneity, $p(\sigma)$. **d.** Same as **c.** but varying degrees of center concentration, $p(\mu)$. **e.** Example functions output by populations with varying degrees of tuning width heterogeneity ($\alpha = 1$: greatest heterogeneity, $\alpha \to \infty$: no heterogeneity) before training (left) and after (right, scatter points = observations). **f.** Same as **d.** but for varying degrees of concentration of tuning centers ($\lambda = 0$: no concentration, uniform tuning; $\lambda = 1$: all tuning concentrated around the center of the stimulus space).

P, target functions, and population tuning parameters,

$$\varepsilon_g(P) = \langle \varepsilon_g(\mathcal{D}, y^*) \rangle_{p(y^*, \mathcal{D}, \Omega)}\,, \tag{1}$$

In particular, we want to assess how tuning parameters $p(\Omega)$ should be shaped with respect to the distribution over target functions $p(y^*)$ so that the network achieves the lowest generalization error. For now, we leave $p(y^*)$ undefined but will return to it after discussing general properties of the population in the next section.

2.2 The role of population geometry, and equivalence to Gaussian Process regression

Optimal weight learning through kernel ridge regression. For a fixed dataset $\mathcal{D}$, the above learning problem is equivalent to kernel ridge regression with optimal weights $\boldsymbol{w}^* = \sqrt{N}(\boldsymbol{\Phi}\boldsymbol{\Phi}^\top + N\gamma \boldsymbol{I})^{-1}\boldsymbol{\Phi}\boldsymbol{y}$, where $\boldsymbol{\Phi}$ is an $N \times P$ matrix with the i-th column equal to $\boldsymbol{r}(\boldsymbol{x}_i; \boldsymbol{\Omega})$, and $\boldsymbol{y} = (y_1, \ldots, y_P)^\top$ is the vector of target outputs. With these weights, we can write the outputs of the network for a test input $\boldsymbol{x}$ as

$$\hat{y}(\boldsymbol{x}) = \boldsymbol{r}(\boldsymbol{x}; \boldsymbol{\Omega})^\top \boldsymbol{w}^* = \boldsymbol{k}(\boldsymbol{x})^\top (\boldsymbol{K} + \gamma \boldsymbol{I})^{-1} \boldsymbol{y}. \tag{2}$$

Here, $\boldsymbol{K} = \frac{1}{N}\boldsymbol{\Phi}^\top \boldsymbol{\Phi}$ is the population kernel Gram matrix with elements given by the kernel function, $[\boldsymbol{K}]_{ij} = K(\boldsymbol{x}_i, \boldsymbol{x}_j) = \frac{1}{N}\boldsymbol{r}(\boldsymbol{x}_i; \boldsymbol{\Omega})^\top \boldsymbol{r}(\boldsymbol{x}_j; \boldsymbol{\Omega})$, and $\boldsymbol{k}(\boldsymbol{x})$ is a P-dimensional vector with elements $[\boldsymbol{k}(\boldsymbol{x})]_i = K(\boldsymbol{x}_i, \boldsymbol{x})$. The learned weights and the network's output are shaped by the population kernel. This observation [25, 26] in part motivates the study of population kernels in neuroscience.

Our interest is in how the population tuning statistics, $p(\Omega)$, shape the population kernel, and thereby the solutions learned by the network. Note that the kernel can be written as

$$K(\boldsymbol{x}_i, \boldsymbol{x}_j) = \frac{1}{N}\boldsymbol{r}(\boldsymbol{x}_i; \boldsymbol{\Omega})^\top \boldsymbol{r}(\boldsymbol{x}_j; \boldsymbol{\Omega}) = \frac{1}{N}\sum_{n=1}^{N} r(\boldsymbol{x}_i; \Omega_n) r(\boldsymbol{x}_j; \Omega_n) \approx \langle r(\boldsymbol{x}_i; \Omega_n) r(\boldsymbol{x}_j; \Omega_n) \rangle_{p(\Omega)}\,, \tag{3}$$

where the last approximation assumes the large population limit, $N \to \infty$. This shows explicitly how tuning statistics shape learned solutions: Tuning shapes the population kernel, which in turn impacts the intrinsic geometry of population activity (Fig 1c). This geometry then shapes the network outputs. In the large population limit, the final approximation in Eq. (3) becomes precise. A natural question is whether the kernel regime is relevant in finitely-sized neural populations. Indeed, we show that the population kernel converges (in terms of relative error) to the large-N analytic kernel even for moderately-sized populations, achieving near perfect alignment [27–29] to the analytic kernel in

populations ranging from $N \sim 10^2 - 10^4$ (Fig. S2), suggesting applicability of this perspective even in rather small biological neural populations.

Equivalence to Gaussian Process regression. We now show that a slightly different construction permits a Bayesian perspective on the learning problem. Specifically, placing a (hyper-)prior over the readout weights induces a *generative model over functions* the network expects to encounter. Furthermore, conditioned on $\mathcal{D}$, the posterior mean of the generative model coincides with the kernel regression solution, permitting the study of network solutions from a Bayesian perspective.

Concretely, we start by placing a hyper-prior over the network's readout weights: $w_n \sim \mathcal{N}(0, \sigma_w^2/N)$. With such hyper-priors, the network outputs converge in the large population limit, $N \to \infty$, to a Gaussian Process (GP) ([30–32]; App. B) with mean and covariance given by

$$\begin{aligned} \langle \hat{y}(\boldsymbol{x}) \rangle_{p(\boldsymbol{w},\Omega)} &= 0, \\ \langle \hat{y}(\boldsymbol{x}) \hat{y}(\boldsymbol{x}') \rangle_{p(\boldsymbol{w},\Omega)} &= \sigma_w^2 \left\langle r(\boldsymbol{x};\Omega_n) r(\boldsymbol{x}';\Omega_n) \right\rangle_{p(\Omega)} = \sigma_w^2 K(\boldsymbol{x},\boldsymbol{x}'). \end{aligned} \tag{4}$$

This kernel $K(\cdot,\cdot)$ equals that of Eq. (3); though, here, it reflects the structure of the network output with *a-priori* weights before they have been adjusted to a particular dataset. Concretely, taking a Bayesian perspective illuminates how tuning statistics shape *priors* over outputs, in addition to learned outputs. Beyond shaping data-dependent solutions as we saw in the previous section, the population kernel can also be seen as instantiating a prior over functions the network expects to encounter (Fig. 1e,f).

To confirm equivalence to kernel ridge regression, we assume that the target outputs y_i are noisy draws from the target function, given by $y_i \sim \mathcal{N}(y^*(\boldsymbol{x}_i), \sigma_S^2)$ (for some unknown function y^*). Then, the GP posterior predictive density of a new input $\boldsymbol{x}$ given a dataset $\mathcal{D}$ is Gaussian distributed with mean

$$\langle \hat{y}(\boldsymbol{x}) | \mathcal{D} \rangle = \boldsymbol{k}(\boldsymbol{x})^\top \left(\boldsymbol{K} + \nu \boldsymbol{I} \right)^{-1} \boldsymbol{y}, \tag{5}$$

matching the kernel ridge regression solution, Eq. (2), with $\gamma = \nu$ (see App. C for details and posterior variance; omitted for brevity). In the above expression, the regularization strength $\nu = \sigma_S^2/\sigma_w^2$ is higher if target outputs are deemed more noisy (higher σ_S^2) and/or trained weights are expected to have smaller magnitudes (stronger shrinkage prior through smaller σ_w^2).

In summary, taking a Bayesian perspective on our problem illuminates the dual roles of population tuning statistics. Firstly, tuning statistics shape the *priors* over function outputs. In addition, these same tuning statistics govern how populations generalize to unseen data after training.

Tuning statistics shape the kernel structure. The population's prior over target functions is determined by the parametric tuning function, $r(\cdot;\cdot)$, and the distribution of the tuning parameters $p(\Omega)$. For example, for bell-shaped tuning curves with $r(\boldsymbol{x};\Omega_n) = \sqrt{a_n} \exp(-\|\boldsymbol{x} - \boldsymbol{\mu}_n\|^2/\sigma_n^2)$ and priors $p(\mu_{n,i}) = \mathcal{U}(-L, L)$ with $L \to \infty$, $p(\sigma_n) = \delta(\ell - \sigma_n)$, and $p(a_n) = \mathcal{U}[g - G, g + G]$, the population kernel becomes a standard radial basis function (RBF) kernel, $K(\boldsymbol{x},\boldsymbol{x}') \propto \exp(-\|\boldsymbol{x} - \boldsymbol{x}'\|^2/2\ell^2)$ (Fig. 1e [$\alpha \to \infty$], App. D.2). A population associated with an RBF kernel outputs functions that vary smoothly on a single length scale. The kernel length-scale parameter ℓ is the shared tuning width across neurons in the population. If, instead, $p(\sigma_n^2) = \text{InvGamma}(\alpha, \alpha\ell^2)$, then the population kernel becomes a rational quadratic (RQ) kernel, $K(\boldsymbol{x},\boldsymbol{x}') \propto \left(1 + \|\boldsymbol{x} - \boldsymbol{x}'\|/(\alpha\ell^2)\right)^{-\alpha}$ (Fig. 1c,e, App. D.3). The linear readout of such a networks supports functions that vary on multiple length scales. The parameter α here controls both the range of supported length scales, as well as the variability of tuning widths across neurons in the population. Note that these are special cases, and our model permits a diverse array of population tuning and thereby population kernels. For example, modulating the distribution of the tuning centers can lead to non-stationary population kernels: populations that concentrate their tuning in a narrow range of the stimulus space induce stimulus-dependent kernel functions. Such populations act as priors over functions with sharp peaks in areas of the stimulus space that correspond to dense tuning (Fig 1f). In general, if the tuning distribution of a particular parameter is a weighted mixture distribution, the population's kernel becomes a weighted mixture of population kernels with fixed parameters (App. D.1)—as illustrated here by the RQ kernel arising from a mixture of RBF kernels with different length scales. The model also supports populations with multimodal tuning curves (App. D.5) and populations with periodic tuning (App. D.6), for which we can still find analytical expressions for the associated kernel functions (see App. D.6). The general expression for the population kernel in Eq. (3) also applies to

more general tuning functions and parameter distributions, such as neurons with asymmetric tuning curves. In these cases, we might not be able to find analytical kernel functions. Nonetheless, all predictions thus far and in the remainder of the paper still apply.

2.3 Function priors that minimize the generalization error

Linking population tuning statistics to GP prior kernels naturally leads to the question of which target function distribution $p(y^*)$ these populations might be optimized to fit. For this we leverage Bayesian decision theory [33], which tells us that the Bayes risk—in our case the generalization error—is minimized if the assumed generative model, including prior and likelihood, matches the data-generating process. For us, this implies that the generalization error is minimized if the student and teacher likelihoods and priors match. The latter is satisfied if the target functions are drawn from a GP whose kernel matches the population kernel $K(\cdot, \cdot)$ instantiated by the population tuning statistics $p(\Omega)$, that is $y^* \sim \mathcal{GP}(0, K)$.

For the remainder of this work we consider multiple consequences of this insight. In Sec. 3, we ask what happens in case of a mismatch between the "teacher" kernel K_T that specifies the prior over target functions, $y^* \sim \mathcal{GP}(0, K_T)$ and the "student" kernel K_S that is determined by the population tuning statistics $p(\Omega)$. In Sec. 4, we ask whether adaptation of neural tuning with experience can be linked to adjusting parameters of the student kernel given some observed training data.

3 Neural population tuning statistics predict generalization error

We first determine the generalization error under the assumption of a matching teacher and student kernels. In this case, the population tuning statistics are optimally tuned to match the prior over target functions. Then, the generalization error is that of the teacher posterior mean, denoted as $\langle y^*(\boldsymbol{x})|\mathcal{D}\rangle$, $\varepsilon_{\text{Bayes}}(P) = \langle(\langle y^*(\boldsymbol{x})|\mathcal{D}\rangle - y^*(\boldsymbol{x}))^2\rangle_{p(y^*,\mathcal{D},\boldsymbol{x})}$, which evaluates to

$$\varepsilon_{\text{Bayes}}(P) = \left\langle K_T(\boldsymbol{x},\boldsymbol{x}) + \text{Tr}\left(\left\langle \boldsymbol{k}_T(\boldsymbol{x})\boldsymbol{k}_T(\boldsymbol{x})^\top \right\rangle_{p(\boldsymbol{x})} (\boldsymbol{K}_T + \sigma_T^2 \boldsymbol{I})^{-1}\right)\right\rangle_{p(\{\boldsymbol{x}_i\},\boldsymbol{x})}, \tag{6}$$

where, as above, the dependence on the dataset size P is made explicit (see App E.1 for details). This error is the irreducible Bayes error arising from noise in the teacher model.

Let us now consider what happens if the student kernel K_S, induced by specific population tuning statistics, does not match the teacher kernel K_T. Then, the generalization error is that of the student's posterior mean $\langle \hat{y}_S(\boldsymbol{x})|\mathcal{D}\rangle$ while assuming that the training data is generated according to the teacher kernel. This generalization error, now $\langle(\langle \hat{y}_S(\boldsymbol{x})|\mathcal{D}\rangle - y^*(\boldsymbol{x}))^2\rangle_{p(y^*,\mathcal{D},\boldsymbol{x})}$, can be decomposed into the above irreducible Bayes error and an additional excess error term that quantifies the increase in generalization error as a result of misalignment between the student and teacher kernels,

$$\varepsilon_g(P) = \varepsilon_{\text{Bayes}}(P) + \varepsilon_{\text{Align}}(P). \tag{7}$$

Here, $\varepsilon_{\text{Align}}(P)$, whose expression is given in App. E.2, quantifies the squared average deviation between the student and teacher posterior means. As a result, $\varepsilon_{\text{Align}}(P) \geq 0$, and $\varepsilon_{\text{Align}}(P) = 0$ only if[1] $K_T = K_S$ and $\sigma_T = \sigma_S$. In this case, $\varepsilon_g(P) = \varepsilon_{\text{Bayes}}(P)$ for all P, in agreement with past work [34]. This implies that any misalignment between the population tuning statistics and the prior over target functions increases the generalization error—an increase we can quantify when both teacher and student kernels are known (Fig. 2c).

Note that all expressions for the generalization error (Eqs. (6), (7) & (66)) implicitly assume that we continue to operate in the large population size limit. As mentioned in the previous section, the empirical population kernel converges to the large N population kernel even for moderately sized populations (Fig. S2). We further confirm that the expressions for generalization error, using empirical population kernels obtained in finitely-sized neural populations, are nearly perfectly predictive in this same moderate-population-size regime. This suggests that our theory also applies to populations with sizes that are biologically plausible (Fig. S2d,h).

[1] If K_T and K_S are diagonalizable in the same basis (e.g., both K_T and K_S are stationary so that both can be diagonalized in a Fourier basis), it is sufficient that they have matching kernel spectral densities rather than equivalent functional forms [34] More generally, if the kernels do not admit identical eigenbases, the stricter condition that the kernels are equivalent holds (App. E.2).

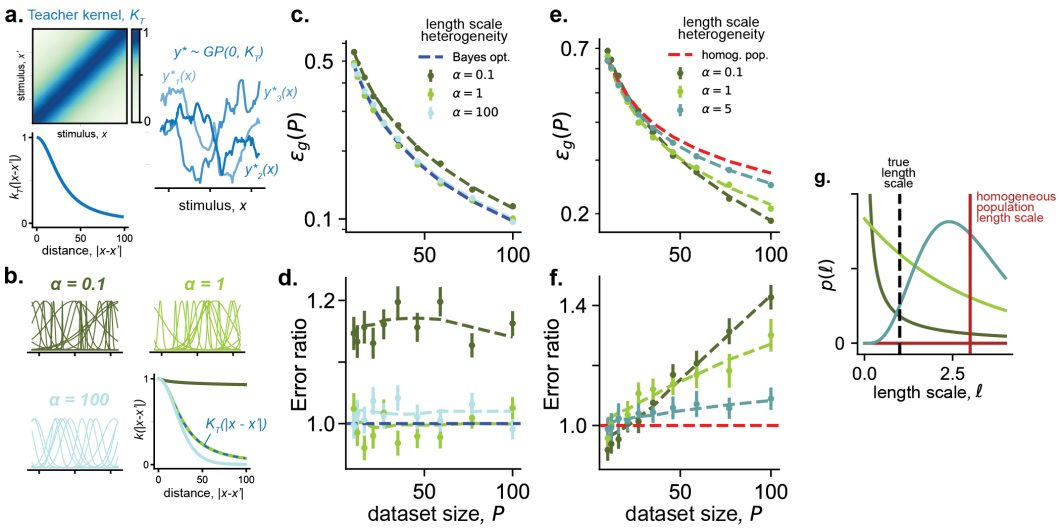

Figure 2: **Population tuning statistics predict generalization error. a.** Illustration of the teacher kernel (top left), the kernel function (bottom left) and sample target functions drawn from the teacher GP (right panel). **b.** Neural populations with varying degrees of tuning width heterogeneity ($\alpha = 0.1$: most heterogeneity, $\alpha = 100$: least), and corresponding kernel functions (bottom right panel). The population with heterogeneity $\alpha = 1$ induces a population kernel that matches the teacher. **c.** Generalization error as a function of dataset size across tested populations (dots = simulations, dashed lines = theory). **d.** Generalization error ratio relative to Bayes optimal error ($\varepsilon_g(P)/\varepsilon_{\text{Bayes}}(P)$, higher = worse). **e.** Comparison of generalization error between heterogeneous populations $\alpha = \{0.1, 1.0, 5.0\}$ versus a homogeneous population, both with mismatched length-scales to the teacher. **f.** Generalization error ratio of homogeneous population relative to heterogeneous populations ($\varepsilon_g^{\text{hom.}}(P)/\varepsilon_g^{\text{het.}}(P)$, higher = het. population better). **g.** Illustration of how heterogeneous populations implicitly support the teacher kernel (black dashed line), unlike the homogeneous population (red line). All error bars = ±SEM.

This leads to several interesting predictions about neural population tuning statistics. In particular, it provides a novel interpretation for the benefit of heterogeneously tuned populations—for example, populations in which tuning curves vary in widths across neurons. To illustrate this, consider a family of target functions that smoothly vary on multiple length-scales. Such target functions can be sampled from a GP with scale mixture RBF kernel, $K_T(\boldsymbol{x}, \boldsymbol{x}') \propto \int K^{\text{RBF}}(\boldsymbol{x}, \boldsymbol{x}'; \sigma) p(\sigma) d\sigma$, where $p(\sigma)$ is the mixing distribution of the length-scales. A neural population that achieves optimal generalization error with such target functions is one whose tuning statistics induce a matching kernel—in this case, a population with Gaussian tuning curves, $r(\boldsymbol{x}; \Omega_n) \propto \exp(-\|\boldsymbol{x} - \boldsymbol{\mu}_n\|^2 / 2\sigma_n^2)$ with uniform tuning centers $p(\mu)$, and a distribution $p(\sigma)$ over tuning widths that matches the mixing distribution of the teacher (Fig. 2a,b). Any mismatch between length-scale and tuning width distributions causes an increase in the generalization error (Fig. 2c,d).

In addition, our work suggests that tuning heterogeneity might reflect the population's uncertainty about the true corresponding parameter in the distribution of target functions. Implementing such uncertainty through tuning heterogeneity can mitigate errors incurred through model mismatch. For example, consider the case where K_T is an RBF kernel with length-scale ℓ. All target functions sampled from this kernel vary on this length scale. Now, consider two neural populations: a homogeneous one in which all neurons share the same tuning curve width, $\hat{\sigma} \neq \ell$, and a heterogeneous one where these widths vary across neurons according to some $p(\sigma)$ with mean $\langle \sigma_n \rangle = \hat{\sigma}$. As Fig. 2e,f illustrates, the homogeneous population has a systematically worse generalization error than heterogeneous one (Fig. 2e,f). This is because the teacher kernel's length-scale is within the support of $p(\sigma)$, and thus expressible by the heterogeneous population, but not by the homogeneous population (Fig. 2g). Importantly, for such a teacher kernel, the optimal population would feature homogeneous tuning widths, σ^*, that match the teacher's length scale $\sigma^* = \ell$—something that neither the homogeneous nor the heterogeneous population accomplish. However, unlike the homogeneous

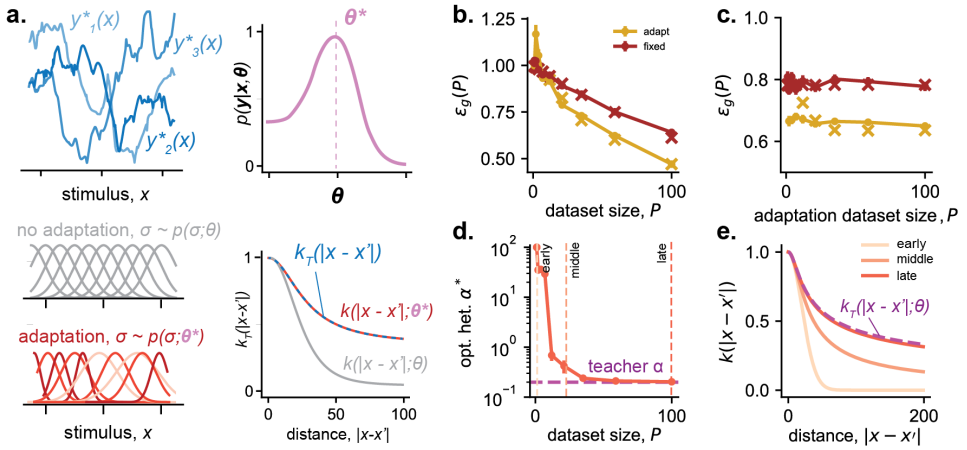

Figure 3: **Tuning adaptation through LML maximization. a.** Schematic of adaptation framework: Sample target functions from teacher GP (top left panel). A population (middle left) undergoes adaptation (bottom left) wherein population statistics adjust to those that maximize the LML (top right). This induces an adaptation in the population statistics and associated kernel (bottom right). **b.** Comparison of generalization error for adapting to a single function sampled from the teacher kernel between a population that adapts tuning to max-LML vs. one that does not (dots = simulations, x = theoretical generalization error for α averaged from runs in **d**; no intermediate dataset size-dependent theory provided, as the optimal α is also dataset-size dependent). **c.** Comparison of generalization error across 50 other functions sampled from teacher kernel (fixed data set size $P = 50$) after LML-driven adaption on function sampled in **b.** for different dataset sizes used for adaptation (horz. axis). **d.** A heterogeneously tuned population achieves max-LML with α's that are dataset size (or experience)-dependent, eventually converging to the true teacher α (dotted line). **e.** Corresponding kernel to early, middle and late amount of experience. All error bars = $\pm$SEM.

population, the heterogeneous population implicitly provides support for the teacher kernel within its associated population kernel.

4 Adaptation of population tuning statistics as model selection

4.1 Population tuning adaptation through maximizing the marginal data likelihood

As we have seen, the generalization error decomposes into the irreducible Bayes error and a second error term arising from tuning mismatch to the target function distribution. Thus, we next asked whether experience could be used to adapt population tuning statistics to reduce this second term.

In GP regression, kernel hyper-parameters $\boldsymbol{\theta}$, which parameterize the kernel function $K(\cdot, \cdot; \boldsymbol{\theta})$, are often adjusted to best explain the data in a process called model selection. A common approach for doing so is through maximizing the log-marginal likelihood (LML), $\log p(\boldsymbol{y}|\boldsymbol{x}, \boldsymbol{\theta})$, of the data:

$$\boldsymbol{\theta}^* = \arg\max_{\boldsymbol{\theta}} \left(-\frac{1}{2}\boldsymbol{y}^\top (\boldsymbol{K}(\boldsymbol{\theta}) + \sigma_S^2 \boldsymbol{I})^{-1}\boldsymbol{y} - \frac{1}{2}\log|\boldsymbol{K}(\boldsymbol{\theta}) + \sigma_S^2\boldsymbol{I}| - \frac{P}{2}\log(2\pi) \right). \tag{8}$$

An RBF kernel, for example, which corresponds to a population of homogeneously tuned Gaussian tuning curves with shared widths σ, has one such parameter $\theta = \sigma$. Then, optimizing the LML corresponds to identifying the shared tuning width that best explains observations. More generally, LML maximization is an appealing objective to adapt population tuning as (i) the LML's Gaussian structure makes it easy to compute and (ii) it prevents overfitting by implicitly balancing model fit (first term in Eq.(8)) with model complexity (second term, Eq.(8)). To better understand why maximizing the LML helps reduce the error arising from kernel mismatch, we observe that the LML of the teacher is always greater than or equal to that of the student. Furthermore, their difference averaged over datasets of a fixed size is the average KL divergence between their respective marginal likelihoods (App. F). Maximizing the student population's LML through tuning adaptation should

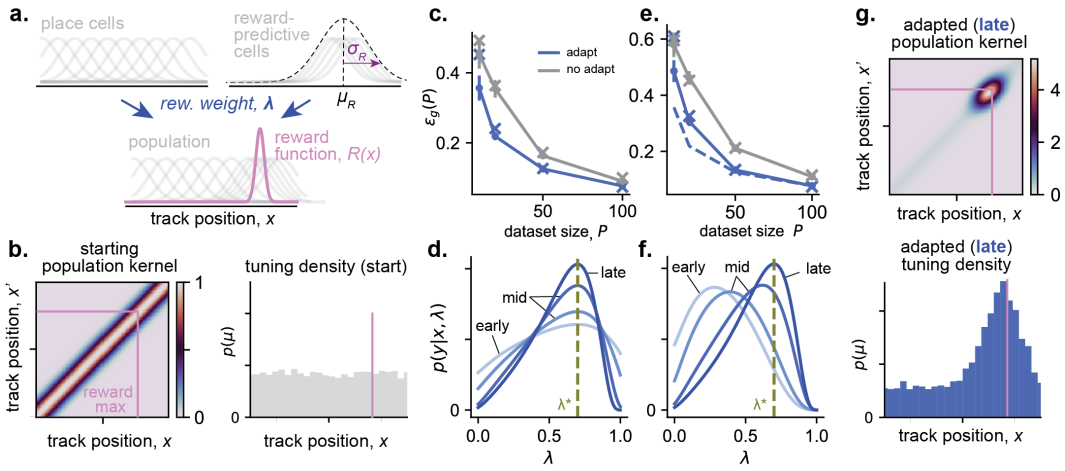

Figure 4: **Reward-predictive tuning concentration as model-selection-driven adaptation**. **a.** Schematic of model population (mixture of overall population tuning controlled by reward weight parameter λ) and reward function. **b.** Before any exposure to the environment, the population's tuning centers are uniformly distributed. **c.** Generalization error in a population with a properly calibrated noise model as its population encounters samples from the target reward function, it either undergoes adaptation (blue line, dots = simulation, x = theory with corresponding optimized λ) or does not (gray line, dots = simulation, x = theory with fixed λ). **d.** The evolution of the LML-maximizing tuning mixture over the course of learning. **e.** Same as **c.** but with a titrated noise model. **f.** Same is **d.** but with a titrated noise model (dotted line = adapted generalization error from **c.**). Here, the peak shifts from uniform ($\lambda = 0$) to a mixture of concentrated and distributed tuning ($\lambda = \lambda^*$). **g.** The converged population kernel (upper) later in learning ($P = 100$ in both **c.** and **e.**) and corresponding tuning density (lower). All error bars = $\pm$SEM obscured by dots (no error for dashed trace in **e**.).

thus decrease this divergence, thereby lowering the generalization error, which is indeed minimized if the population tuning statistics match the true generative model of target functions.

4.2 Adaptation of population statistics improves generalization error

We next explored if adaptation of population tuning statistics by maximizing the LML indeed lowers the generalization error. To do so, we fixed a teacher kernel K_T matching that of Fig. 2a with parameter α that controls the distribution of supported length-scales. We then compared the generalization error of two neural populations with matching kernels and initially mistuned α: one that adapted the tuning statistics controlled by α by maximizing the LML, and another that undergoes no adaptation (Fig. 3a). We found that tuning adaptation is experience-dependent and results in a lower generalization error (Fig. 3b,c). In our experiments, we observed convergence of the population's α to the teacher's corresponding value with increasing dataset size (Fig. 3d,e). Interestingly, while the LML-maximizing population α did not match the teacher's for small dataset sizes (Fig. 3d), adjusted tuning by LML-maximization nonetheless reduced the generalization error early in training (Fig. 3b,c). This suggests that population tuning adaptation is advantageous even early in learning without substantial knowledge of the teacher model, and that with sufficient experience, tuning statistics converge to a stable state (Fig. 3d).

4.3 Reward-associated spatial representation adaptation through model selection

Recent experiments have demonstrated that in the hippocampal spatial code with otherwise uniformly distributed tuning, a dedicated subset of "reward-associated cells" adapt with experience to progressively over-represent rewarding zones in a linear track environment with a fixed reward location [12, 35]. The reason for this over-representation—and in particular, its computational role—remains poorly understood (though see [36]).

We asked whether the observed over-representation could be explained by assuming that a place field population with a reward-representing subpopulation adapts to minimize its generalization error

when predicting reward functions. To do so, we constructed a model population whose Gaussian tuning curves (uniform widths, for simplicity) tiled a linear track, and whose task was to learn a highly localized reward function that peaks at a distal location on the track (Fig. 4a). The model population had (i) one typical place cell subpopulation whose tuning centers uniformly tiled the track, and (ii) another reward-predictive subpopulation whose tuning centers concentrated around a particular location μ_R in the environment (Fig. 4a). This led to the tuning center distribution $p(\mu) = \lambda\mathcal{N}(\mu_R, \sigma_R^2) + (1-\lambda)\mathcal{U}(-L, L)$ with mixing weight $\lambda \in [0, 1]$ ($\lambda \to 0$: uniform centers; $\lambda \to 1$: concentration around μ_R).

The corresponding population kernel is the weighted combination $K(\lambda, \sigma, \sigma_R, \mu_R) = \lambda K^{\text{RP}}(\sigma, \sigma_R, \mu_R) + (1-\lambda)K^{\text{TP}}(\sigma)$ of a typical place field kernel $K^{\text{TP}}(\sigma)$ (an RBF kernel with length-scale σ), and the reward-predictive subpopulation kernel $K^{\text{RP}}(\sigma, \sigma_R, \mu_R)$ (an RBF kernel modulated by μ_R and σ_R) (Fig. 4b,g, see App. D.4). We next assume that a target reward function $R \sim \mathcal{GP}(0, K_T)$ with $K_T = \lambda^* K^{\text{RP}}(\sigma^*, \sigma_R^*, \mu_R^*) + (1-\lambda^*)K^{\text{TP}}(\sigma^*)$ and Gaussian emissions $y_i \sim \mathcal{N}(R(x_i), \sigma_T^2)$. This model generates realistic, locally peaked reward functions, and also permits a set of ground-truth optimal kernel parameters for the population to achieve.

With a properly calibrated noise model of the student ($\sigma_T = \sigma_S$), LML maximization drove population tuning centers to concentrate around the reward function's peak (Fig. 4d). More experience resulted in lower generalization error than a comparable population with spatially homogeneous tuning (Fig. 4c[2]), a lower LML variance, and LML-maximizing λ's closer to the teacher kernel's λ^*. Notably, this adaptation occurred even with minimal experience. This is in part because maximizing the LML identifies tuning that induces the least complex kernel. We found that a population with concentrated tuning restricted the prior to a more limited set of functions and thus gave rise to a simpler kernel than one with homogeneous tuning (Fig. S3).

Our model revealed an interplay between the magnitude and speed of tuning adaptation in this task and the population's noise model. In particular, if the student's noise model starts with $\sigma_S < \sigma_T$ and is slowly titrated to match the teacher's, tuning curves concentrate more gradually around the reward location with experience (Fig. 4f) without incurring substantial excess error (Fig. 4e). This is because the assumed noise variance determines the amount of regularization imposed by the population (App. D.2), which additionally controls adaptation speed. As a consequence, populations with static and mis-specified noise, $\sigma_T \neq \sigma_S$ either over- or under-represent the rewarding location in the environment (Fig. S4). Such systematic over- or under-representation of the rewarding location may be in part attributed to trade-offs implicitly selected for in LML maximization. With small regularization (σ_s small), a more flexible RBF kernel better explains small variations in data and, because of limited regularization, the data-explaining term of the LML outweighs the complexity regularization term, driving the population to exhibit less localized representations. Conversely, with greater regularization ($\sigma_S \to \sigma_T$), data is sufficiently well-explained by the more restrictive envelope kernel emerging from a dense reward representation, and the kernel complexity term of the LML outweighs the data-explaining term, driving more localized representations. More generally, our framework illuminates how the magnitude and dynamics of tuning adaptation in the reward location depends on population tuning parameters, as well as the population's assumed model of the environment, such as the amount of noise it expects in outputs from the teacher (Fig. 4e,f).

5 Discussion

Heterogeneous tuning is ubiquitous in neural populations. While its role in encoding precision is well studied, its implications for learning and generalization are less understood. In this work, we proposed a probabilistic model that links a population's tuning statistics to its generalization error across a distribution of target functions. Our model allowed us to quantify the excess error under task-tuning mismatch, when tuning statistics are poorly calibrated for the encountered distribution of tasks. More broadly, our results suggest an interpretation of tuning heterogeneity as uncertainty over task-relevant function parameters (e.g., characteristic length scales). Finally, the same probabilistic perspective yielded a normative objective for how neural populations can resolve poorly calibrated priors. Indeed, adapting tuning statistics by maximizing the marginal data likelihood reduces

[2]Computing the theoretical generalization error in this experiment requires using error expressions that contain the full student and teacher kernels (Eq. (66)) rather than simply their respective spectra, as in [34]. This is required, as the student and teacher kernels are not necessarily diagonalizable in the same eigenbasis when their respective tuning mixture λ parameters are different.

generalization error relative to non-adapting populations. We further applied our theory to reward learning in the hippocampus, showing that it accounts for the concentration of reward-predictive spatial representations under asymmetric rewards.

Past theories of learning and generalization in large biological neural networks posited that a population's induced kernel predicts its generalization error in a fixed task [26]. Here, we extend this view to distributions across tasks via a probabilistic connection between the population kernel and its resultant prior over functions. Because tuning variability corresponds to an induced GP prior over functions, we correspondingly model task distributions using GPs. Although this may seem restrictive, GPs capture a broad range of input-output mappings. Nonetheless, our current analysis relies on the assumption that neuronal responses are parametric and deterministic, while real responses are noisy and history-dependent. Future work can assess how stochastic spiking reshapes induced function priors, generalization predictions, and adaptation dynamics.

Efficient coding theories propose that limited neural resources are allocated to prioritize encoding behaviorally-relevant stimuli, predicting tuning diversity that is task-specific [4, 15, 17, 19–21, 23]. Our findings extend this perspective from encoding stimuli to learning arbitrary input-output mappings. Misalignment between task distributions and the populations tuning-induced prior over them causes an increase in generalization error. We propose a role for adaptation in learning, where data-driven adaptation that maximizes the LML objective is analogous to tuning adaptation predicted to arise through the encoding objective in the efficient coding framework. Although we do not impose explicit metabolic constraints, marginal likelihood implicitly trades off data fit and model complexity, which may play a similar regularizing role.

While we propose a normative objective for adjusting tuning statistics to reduce generalization error, it is unclear what biological mechanisms could implement such population-level updates. Related mechanistic work has studied this question applied to reward-driven hippocampal reorganization, but only found such reorganization if model populations were sufficiently small [36]. This result is consistent with the lazy (or kernel learning) regime in which feature learning is suppressed—a well-known feature of gradient descent learning in over-parametrized networks [30, 31, 37]. Work that focuses on biologically plausible update mechanisms that operate on behaviorally relevant timescales [12, 24, 38] and which implement the adaptations described here in over-parameterized neural networks is a promising avenue for future research. Further, our analysis suggests that regularization might play a key role in shaping the dynamics of reward-sensitive adaptation. Work that identifies biological correlates of such regularization would be an exciting avenue for future investigation.

Likelihood maximization is a standard Bayesian criterion for model selection [32]. In our setting, LML maximization selects tuning statistics that reduce generalization error. This echoes past work linking higher evidence to faster learning speeds [39]. However, LML maximization can also fail in some instances, leading to strictly *worse* generalization error—an issue that can be resolved through careful data sub-selection strategies [40]. A natural extension to our framework is to study analogous strategies in the context of population tuning adaptation.

While we restrict our framing and analyses to the study of biological neural systems in this work, we anticipate that some of the insights we develop here may be applicable to the broader machine learning community. Indeed, we cast evidence maximization as a meta-learning strategy in large, over-paramterized neural networks. Future work could investigate if task- and LML-driven meta-learning loops for the systematic shaping of how deep network parameters are initialized may improve low-sample generalization error in deep learning models.

Broader impacts. Overall, our work proposes a new theory of the role of tuning heterogeneity and adaptation in learning. We expect that the broader impacts of this work will be positive, casting tuning heterogeneity within an explainable normative framework. We do not expect any negative societal impacts to emerge as a consequence of this insight.

Acknowledgments and Disclosure of Funding

We thank Camille Rullán Buxó for comments on an earlier version of the manuscript and the Drugowitsch Lab for feedback on the work. This work was supported by a Kempner Graduate Fellowship (Z.C.) and NIH funding (U19NS118246 & R01NS116753; J.D.). Z.C. and J.D. conceived of the study. Z.C. performed the analysis. Z.C. and J.D. interpreted the results. Z.C. and J.D. wrote

and revised the manuscript. The authors have no conflicts of interest to report. Code and data used to produce the manuscript figures is available at https://doi.org/10.5281/zenodo.23984835 [41].

References

[1] Robbe L.T. Goris, Eero P. Simoncelli, and J. Anthony Movshon. Origin and function of tuning diversity in macaque visual cortex. *Neuron*, 88(4):819–831, November 2015. ISSN 0896-6273. doi: 10.1016/j.neuron.2015.10.009. URL http://dx.doi.org/10.1016/j.neuron.2015.10.009.

[2] Benjamin Scholl, Andrew YY Tan, Joseph Corey, and Nicholas J Priebe. Emergence of orientation selectivity in the mammalian visual pathway. *Journal of Neuroscience*, 33(26): 10616–10624, 2013.

[3] Dario L Ringach, Robert M Shapley, and Michael J Hawken. Orientation selectivity in macaque v1: diversity and laminar dependence. *Journal of neuroscience*, 22(13):5639–5651, 2002.

[4] H. B. Barlow, R. M. Hill, and W. R. Levick. Retinal ganglion cells responding selectively to direction and speed of image motion in the rabbit. *The Journal of Physiology*, 173(3): 377–407, September 1964. ISSN 1469-7793. doi: 10.1113/jphysiol.1964.sp007463. URL http://dx.doi.org/10.1113/jphysiol.1964.sp007463.

[5] David G. Clark, L. F. Abbott, and Haim Sompolinsky. Symmetries and continuous attractors in disordered neural circuits. *Nature Neuroscience*, October 2026. ISSN 1546-1726. doi: 10.1038/s41593-026-02452-4. URL http://dx.doi.org/10.1038/s41593-026-02452-4.

[6] P. Dylan Rich, Hua-Peng Liaw, and Albert K. Lee. Large environments reveal the statistical structure governing hippocampal representations. *Science*, 345(6198):814–817, August 2014. ISSN 1095-9203. doi: 10.1126/science.1255635. URL http://dx.doi.org/10.1126/science.1255635.

[7] Jae Sung Lee, John J. Briguglio, Jeremy D. Cohen, Sandro Romani, and Albert K. Lee. The statistical structure of the hippocampal code for space as a function of time, context, and value. *Cell*, 183(3):620–635.e22, October 2020. ISSN 0092-8674. doi: 10.1016/j.cell.2020.09.024. URL http://dx.doi.org/10.1016/j.cell.2020.09.024.

[8] Huanqiu Zhang, P. Dylan Rich, Albert K. Lee, and Tatyana O. Sharpee. Hippocampal spatial representations exhibit a hyperbolic geometry that expands with experience. *Nature Neuroscience*, 26(1):131–139, December 2023. ISSN 1546-1726. doi: 10.1038/s41593-022-01212-4. URL http://dx.doi.org/10.1038/s41593-022-01212-4.

[9] Bruce Harland, Marco Contreras, Madeline Souder, and Jean-Marc Fellous. Dorsal ca1 hippocampal place cells form a multi-scale representation of megaspace. *Current Biology*, 31 (10):2178–2190.e6, May 2021. ISSN 0960-9822. doi: 10.1016/j.cub.2021.03.003. URL http://dx.doi.org/10.1016/j.cub.2021.03.003.

[10] Tamir Eliav, Shir R. Maimon, Johnatan Aljadeff, Misha Tsodyks, Gily Ginosar, Liora Las, and Nachum Ulanovsky. Multiscale representation of very large environments in the hippocampus of flying bats. *Science*, 372(6545), May 2021. ISSN 1095-9203. doi: 10.1126/science.abg4020. URL http://dx.doi.org/10.1126/science.abg4020.

[11] John O'Keefe and Lynn Nadel. *The Hippocampus as a Cognitive Map*. Oxford: Clarendon Press, 1978. ISBN 0-19-857206-9. URL http://hdl.handle.net/10150/620894.

[12] Jeffrey L. Gauthier and David W. Tank. A dedicated population for reward coding in the hippocampus. *Neuron*, 99(1):179–193.e7, July 2018. ISSN 0896-6273. doi: 10.1016/j.neuron.2018.06.008. URL http://dx.doi.org/10.1016/j.neuron.2018.06.008.

[13] Zach Cohen and Jan Drugowitsch. A unifying theory of receptive field heterogeneity predicts hippocampal spatial tuning. *bioRxiv*, July 2025. doi: 10.1101/2025.07.26.666958. URL http://dx.doi.org/10.1101/2025.07.26.666958.

[14] Kukjin Kang, Robert M. Shapley, and Haim Sompolinsky. Information tuning of populations of neurons in primary visual cortex. *The Journal of Neuroscience*, 24(15):3726–3735, April 2004. ISSN 1529-2401. doi: 10.1523/jneurosci.4272-03.2004. URL http://dx.doi.org/10.1523/JNEUROSCI.4272-03.2004.

[15] H B Barlow and W A Rosenblith. *Possible principles underlying the transformations of sensory messages*, pages 217–234. MIT Press, 1961.

[16] Simon Laughlin. A simple coding procedure enhances a neuron's information capacity. *Zeitschrift für Naturforschung C*, 36(9–10):910–912, October 1981. ISSN 0939-5075. doi: 10.1515/znc-1981-9-1040. URL http://dx.doi.org/10.1515/znc-1981-9-1040.

[17] Bruno A. Olshausen and David J. Field. Sparse coding with an overcomplete basis set: A strategy employed by v1? *Vision Research*, 37(23):3311–3325, December 1997. ISSN 0042-6989. doi: 10.1016/s0042-6989(97)00169-7. URL http://dx.doi.org/10.1016/S0042-6989(97)00169-7.

[18] Michael S. Lewicki. Efficient coding of natural sounds. *Nature Neuroscience*, 5(4):356–363, March 2002. ISSN 1546-1726. doi: 10.1038/nn831. URL http://dx.doi.org/10.1038/nn831.

[19] Il Memming Park and Jonathan W. Pillow. Bayesian efficient coding. *bioRxiv*, August 2017. doi: 10.1101/178418. URL http://dx.doi.org/10.1101/178418.

[20] Zhuo Wang, Xue-Xin Wei, Alan A Stocker, and Daniel D Lee. Efficient neural codes under metabolic constraints. In D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc., 2016. URL https://proceedings.neurips.cc/paper_files/paper/2016/file/e6c2dc3dee4a51dcec3a876aa2339a78-Paper.pdf.

[21] Deep Ganguli and Eero P. Simoncelli. Efficient sensory encoding and bayesian inference with heterogeneous neural populations. *Neural Computation*, 26(10):2103–2134, October 2014. ISSN 1530-888X. doi: 10.1162/neco_a_00638. URL http://dx.doi.org/10.1162/NECO_a_00638.

[22] Luke Rast and Jan Drugowitsch. Adaptation properties allow identification of optimized neural codes. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 1142–1152. Curran Associates, Inc., 2020.

[23] Thomas E. Yerxa, Eric Kee, Michael R. DeWeese, and Emily A. Cooper. Efficient sensory coding of multidimensional stimuli. *PLOS Computational Biology*, 16(9):e1008146, September 2020. ISSN 1553-7358. doi: 10.1371/journal.pcbi.1008146. URL http://dx.doi.org/10.1371/journal.pcbi.1008146.

[24] Mayank R. Mehta, Carol A. Barnes, and Bruce L. McNaughton. Experience-dependent, asymmetric expansion of hippocampal place fields. *Proceedings of the National Academy of Sciences*, 94(16):8918–8921, August 1997. ISSN 1091-6490. doi: 10.1073/pnas.94.16.8918. URL http://dx.doi.org/10.1073/pnas.94.16.8918.

[25] Carsen Stringer, Marius Pachitariu, Nicholas Steinmetz, Matteo Carandini, and Kenneth D. Harris. High-dimensional geometry of population responses in visual cortex. *Nature*, 571(7765):361–365, June 2019. ISSN 1476-4687. doi: 10.1038/s41586-019-1346-5. URL http://dx.doi.org/10.1038/s41586-019-1346-5.

[26] Blake Bordelon and Cengiz Pehlevan. Population codes enable learning from few examples by shaping inductive bias. *eLife*, 11, December 2022. ISSN 2050-084X. doi: 10.7554/elife.78606. URL http://dx.doi.org/10.7554/eLife.78606.

[27] Nello Cristianini, John Shawe-Taylor, André Elisseeff, and Jaz S. Kandola. On kernel-target alignment. In *Neural Information Processing Systems*, 2001. URL https://api.semanticscholar.org/CorpusID:17466014.

[28] Simon Kornblith, Mohammad Norouzi, Honglak Lee, and Geoffrey Hinton. Similarity of neural network representations revisited. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 3519–3529. PMLR, 09–15 Jun 2019. URL https://proceedings.mlr.press/v97/kornblith19a.html.

[29] Corinna Cortes, Mehryar Mohri, and Afshin Rostamizadeh. Algorithms for learning kernels based on centered alignment. *J. Mach. Learn. Res.*, 13(1):795–828, March 2012. ISSN 1532-4435.

[30] Christopher K. I. Williams. Computing with infinite networks. In *Proceedings of the 10th International Conference on Neural Information Processing Systems*, NIPS'96, page 295–301, Cambridge, MA, USA, 1996. MIT Press.

[31] Radford M. Neal. *Priors for Infinite Networks*, page 29–53. Springer New York, 1996. ISBN 9781461207450. doi: 10.1007/978-1-4612-0745-0_2. URL http://dx.doi.org/10.1007/978-1-4612-0745-0_2.

[32] Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian Processes for Machine Learning*. The MIT Press, November 2005. ISBN 9780262256834. doi: 10.7551/mitpress/3206.001.0001. URL http://dx.doi.org/10.7551/mitpress/3206.001.0001.

[33] James O. Berger. *Statistical Decision Theory and Bayesian Analysis*. Springer New York, 1985. ISBN 9781475742862. doi: 10.1007/978-1-4757-4286-2. URL http://dx.doi.org/10.1007/978-1-4757-4286-2.

[34] Peter Sollich. Gaussian process regression with mismatched models. In T. Dietterich, S. Becker, and Z. Ghahramani, editors, *Advances in Neural Information Processing Systems*, volume 14. MIT Press, 2001. URL https://proceedings.neurips.cc/paper_files/paper/2001/file/d68a18275455ae3eaa2c291eebb46e6d-Paper.pdf.

[35] Charline Tessereau, Feng Xuan, Zehua Chen, Jack R. Mellor, Peter Dayan, and Daniel Dombeck. Navigating uncertainty: reward location variability induces reorganization of hippocampal spatial representations. *bioRxiv*, January 2025. doi: 10.1101/2025.01.06.631465. URL http://dx.doi.org/10.1101/2025.01.06.631465.

[36] M Ganesh Kumar, Blake Bordelon, Jacob A Zavatone-Veth, and Cengiz Pehlevan. A model of place field reorganization during reward maximization. In Aarti Singh, Maryam Fazel, Daniel Hsu, Simon Lacoste-Julien, Felix Berkenkamp, Tegan Maharaj, Kiri Wagstaff, and Jerry Zhu, editors, *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pages 31892–31929. PMLR, 13–19 Jul 2025. URL https://proceedings.mlr.press/v267/kumar25a.html.

[37] Arthur Jacot, Franck Gabriel, and Clément Hongler. Neural tangent kernel: convergence and generalization in neural networks. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, NIPS'18, page 8580–8589, Red Hook, NY, USA, 2018. Curran Associates Inc.

[38] Elaine Tring, Mario Dipoppa, and Dario L. Ringach. A power law describes the magnitude of adaptation in neural populations of primary visual cortex. *Nature Communications*, 14 (1), December 2023. ISSN 2041-1723. doi: 10.1038/s41467-023-43572-w. URL http://dx.doi.org/10.1038/s41467-023-43572-w.

[39] Clare Lyle, Lisa Schut, Robin Ru, Yarin Gal, and Mark van der Wilk. A bayesian perspective on training speed and model selection. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 10396–10408. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/75a7c30fc0063c4952d7eb044a3c0897-Paper.pdf.

[40] Sanae Lotfi, Pavel Izmailov, Gregory Benton, Micah Goldblum, and Andrew Gordon Wilson. Bayesian model selection, the marginal likelihood, and generalization. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato, editors, *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of*

Machine Learning Research, pages 14223–14247. PMLR, 17–23 Jul 2022. URL https://proceedings.mlr.press/v162/lotfi22a.html.

[41] Zach Cohen. Drugowitschlab/tuningstatisticsaspriors: Camera- ready code, November 2026. URL https://doi.org/10.5281/zenodo.24013238.

[42] C.M. Bishop. *Pattern Recognition and Machine Learning by Christopher M. Bishop*. Springer Science+Business Media, LLC, 2006. URL https://books.google.com/books?id=Y44SyAEACAAJ.

[43] Peter Sollich. Learning curves for gaussian processes. In M. Kearns, S. Solla, and D. Cohn, editors, *Advances in Neural Information Processing Systems*, volume 11. MIT Press, 1998. URL https://proceedings.neurips.cc/paper_files/paper/1998/file/5cbdfd0dfa22a3fca7266376887f549b-Paper.pdf.

Appendix

A Notation table

The below table summarizes the notation used in the paper.

Symbol	Meaning	Notes
D	Input dimensionality	
N	Population size	
$\boldsymbol{x}$	Input, stimulus, latent value	$\boldsymbol{x} \in \mathbb{R}^D$
Ω_n	Tuning parameters of neuron n	$\Omega_n \sim p(\Omega)$, Params. assumed i.i.d.
$\boldsymbol{\Omega}$	Tuning params. whole population	Params. i.i.d.
$\mathcal{D}$	Dataset	$\lvert\mathcal{D}\rvert = P$
$r(\boldsymbol{x}; \Omega_n)$	Tuning function, neuron n	$r : \mathbb{R}^D \to \mathbb{R}$
σ_n	Bandwidth of neuron n	$\sigma_n \sim p(\sigma)$
$\boldsymbol{r}(\boldsymbol{x})$	Population response to $\boldsymbol{x}$	$\boldsymbol{r}(\boldsymbol{x}) \in \mathbb{R}^N$
$\boldsymbol{\Phi}$	Responses to dataset $\mathcal{D}$	$\boldsymbol{\Phi} \in \mathbb{R}^{N \times P}$
$\boldsymbol{w}$	Readout weights	$\boldsymbol{w} \in \mathbb{R}^N$
$\hat{y}(\boldsymbol{x}), \hat{y}_S(\boldsymbol{x})$	Network output ($\cdot_S$ = student)	$\hat{y}(\boldsymbol{x}) = \boldsymbol{r}(\boldsymbol{x})^\top \boldsymbol{w}$
$y^*(\boldsymbol{x})$	Teacher output	
$K(\cdot, \cdot)$	Kernel function	$K : \mathbb{R}^D \times \mathbb{R}^D \to \mathbb{R}$
$K_S(\cdot, \cdot)$	Student kernel	
$K_T(\cdot, \cdot)$	Teacher kernel	
$\boldsymbol{K}_{S,T}$	Student, teacher kernel matrix	Kernel function for all input pairs
$\boldsymbol{k}(\boldsymbol{x})$	Test input kernel	$\boldsymbol{k}(\boldsymbol{x}) \in \mathbb{R}^P, [\boldsymbol{k}(\boldsymbol{x})]_i = K(\boldsymbol{x}_i, \boldsymbol{x})$
σ_S, σ_T	Student, teacher observation noise	
$\varepsilon_g(P)$	Generalization error, P inputs	Eq. (1)
InvGamma$(\alpha, \alpha\ell^2)$	Inverse Gamma distribution	Shape α, scale $\alpha\ell^2$
$\mathcal{U}[g - G, g + G]$	Uniform distribution	Bounds set so avg. = g, $G \leq g$.

B In the large N limit, single-layer neural networks become Gaussian processes

The original derivation for this claim can be found in [31]. For completeness, we provide the derivation here as well.

Assume that we have a single-layer network of N neurons where the n-th neuron has the tuning curve $r(\boldsymbol{x}; \Omega_n)$, where $\boldsymbol{x}$ is a possibly multi-dimensional input to the network and $\Omega_n \sim p(\Omega)$ are the parameters of the neuron n, describing its location in the environment, its peak firing-rate, its width, etc. The network outputs

$$\hat{y}(\boldsymbol{x}_i) = \sum_{n=1}^{N} w_n r(\boldsymbol{x}_i; \Omega_n) = \boldsymbol{r}(\boldsymbol{x}_i; \boldsymbol{\Omega})^\top \boldsymbol{w}, \tag{9}$$

where $\boldsymbol{w} = (w_1, \ldots, w_N)^\top$ is the weight vector from each neuron in the population onto the output. Assume that the weights to the output are i.i.d. initialized with samples from a Gaussian distribution, $w_i \sim \mathcal{N}(0, \sigma_w^2)$. Under this setup, as $N \to \infty$, we will show that $\hat{y}(\cdot)$ can be viewed as a sample from a Gaussian process with a kernel, $K(\boldsymbol{x}_i, \boldsymbol{x}_j) = \langle r(\boldsymbol{x}_i; \Omega_n) r(\boldsymbol{x}_j; \Omega_n) \rangle_{p(\Omega)}$. Under this interpretation, the stochastic parameters of the tuning curves in the population induce a GP prior over output mappings from the population.

To see that this is the case, first note that as $N \to \infty$, by the Central Limit Theorem (CLT), $\hat{y}(\boldsymbol{x}_i)$ becomes Gaussian distributed. To see this, note that $\hat{y}(\boldsymbol{x}_i)$ is a random variable; the sum of the product of i.i.d. Gaussian random variables with the outputs of the population neurons. The parameters of

each neuron are also i.i.d., meaning that the contribution of each neuron times its weight is i.i.d. The mean of the contribution is $\langle w_n r(\boldsymbol{x}_i;\Omega_n)\rangle_{p(w,\Omega)} = \langle w_n\rangle_{p(w)}\langle r(\boldsymbol{x}_i;\Omega_n)\rangle_{p(\Omega)} = 0$. The variance is given by $\left\langle (w_n r(\boldsymbol{x}_i;\Omega_n))^2\right\rangle_{p(w,\Omega)} = \left\langle w_n^2\right\rangle_{p(w)}\left\langle r^2(\boldsymbol{x}_i,\Omega_n)\right\rangle_{p(\Omega)} = \sigma_w^2\left\langle r^2(\boldsymbol{x}_i,\Omega_n)\right\rangle_{p(\Omega)}$. Define $\sigma_{\boldsymbol{x}_i} = \left\langle r^2(\boldsymbol{x}_i,\Omega_n)\right\rangle_{p(\Omega)}$ for notational simplicity, and because the variance is the same for each neuron n. Then by the CLT,

$$\hat{y}(\boldsymbol{x}_i) \sim \mathcal{N}\left(0, N\sigma_w^2\sigma_{\boldsymbol{x}_i}\right). \tag{10}$$

To prevent the variance from growing unbounded, we note the need to set the output weights to $w_n/\sqrt{N}$ such that $\hat{y}(\boldsymbol{x}_i) \sim \mathcal{N}\left(0, \sigma_w^2\sigma_{\boldsymbol{x}_i}\right)$.

Next we consider the joint distribution of $\hat{y}(\boldsymbol{x}_1),\ldots,\hat{y}(\boldsymbol{x}_P)$ for some collection of P inputs. Considered as a vector, we can see that like above, this vector is the sum of infinite vectors with i.i.d components,

$$\begin{bmatrix}\hat{y}(\boldsymbol{x}_1)\\ \vdots \\ \hat{y}(\boldsymbol{x}_P)\end{bmatrix} = \begin{bmatrix}\sum_n w_n r(\boldsymbol{x}_1;\Omega_n)\\ \vdots \\ \sum_n w_n r(\boldsymbol{x}_P;\Omega_n)\end{bmatrix}. \tag{11}$$

So, following the same logic as above, the multidimensional CLT (and proper scaling on the weights) tells us that as $N\to\infty$,

$$\begin{bmatrix}\hat{y}(\boldsymbol{x}_1)\\ \vdots \\ \hat{y}(\boldsymbol{x}_P)\end{bmatrix} \sim \mathcal{N}\left(\mathbf{0}, \begin{bmatrix}\mathrm{Var}(\hat{y}(\boldsymbol{x}_1)) & \ldots & \mathrm{Cov}(\hat{y}(\boldsymbol{x}_1),\hat{y}(\boldsymbol{x}_P))\\ \vdots & \ddots & \vdots \\ \mathrm{Cov}(\hat{y}(\boldsymbol{x}_P),\hat{y}(\boldsymbol{x}_1)) & \ldots & \mathrm{Var}(\hat{y}(\boldsymbol{x}_P))\end{bmatrix}\right) \tag{12}$$

where, with proper scaling,

$$\mathrm{Cov}(\hat{y}(\boldsymbol{x}_i),\hat{y}(\boldsymbol{x}_j)) = \sigma_w^2\left\langle r(\boldsymbol{x}_i;\Omega_n)r(\boldsymbol{x}_j;\Omega_n)\right\rangle_{p(\Omega)} = \sigma_w^2 K(\boldsymbol{x}_i,\boldsymbol{x}_j). \tag{13}$$

This suggests that when initialized, the functions output by the network can be seen as samples from a Gaussian process prior with kernel K, which is just a function of the tuning curves and their parameters, and, in particular, how these parameters vary in the population.

C Comparing kernel regression and GP posterior solutions

Let us use the following simplified notation,

$$\hat{y}(\boldsymbol{x}) = \frac{1}{\sqrt{N}}\sum_{n=1}^{N} w_n r(\boldsymbol{x};\Omega_n) = \frac{1}{\sqrt{N}}\boldsymbol{w}^\top \boldsymbol{r}(\boldsymbol{x}), \tag{14}$$

where we again assume $w_i \sim \mathcal{N}(0,\sigma_w^2)$, and where we introduce the vector notation $\boldsymbol{r}(\boldsymbol{x}) = (r(\boldsymbol{x};\Omega_1),\ldots,r(\boldsymbol{x};\Omega_n))^\top$. This again leads to $\langle\hat{y}(\boldsymbol{x})\rangle = 0$, and

$$\langle\hat{y}(\boldsymbol{x})\hat{y}(\boldsymbol{x}')\rangle = \sigma_w^2\frac{1}{N}\sum_{n=1}^{N} r(\boldsymbol{x};\Omega_n)r(\boldsymbol{x}';\Omega_n) \approx \sigma_w^2\left\langle r(\boldsymbol{x};\Omega_n)r(\boldsymbol{x}';\Omega_n)\right\rangle, \tag{15}$$

where the last approximation assumes $N\to\infty$ dense tuning and a Monte Carlo approximation of the expectation. Thus, we get the same GP kernel as before.

Further, assume a training set $\mathcal{D} = \{(\boldsymbol{x}_p, y_p)\}_{p=1}^P$ with P training samples. We will in turn derive the Kernel regression and GP posterior solutions and compare them.

C.1 Kernel regression solution

Assume the loss function

$$\mathcal{L}(\boldsymbol{w}) = \frac{1}{2P}\sum_{p=1}^{P}\left(\hat{y}(\boldsymbol{x}_p) - y_p\right)^2 + \frac{\gamma}{2}\|\boldsymbol{w}\|^2 = \frac{1}{2P}\left\|\frac{1}{\sqrt{N}}\boldsymbol{\Phi}\boldsymbol{w} - \boldsymbol{y}\right\|^2 + \frac{\gamma}{2}\|\boldsymbol{w}\|^2, \tag{16}$$

where the $P \times N$ matrix $\boldsymbol{\Phi}$ has $\boldsymbol{r}(\boldsymbol{x}_p)^\top$ as its p-th row, and $\boldsymbol{y} = (y_1, \ldots, y_P)^\top$ collects the target values into a vector, such that $\boldsymbol{h} = (h(\boldsymbol{x}_1), \ldots, h(\boldsymbol{x}_P))^\top = \frac{1}{\sqrt{N}}\boldsymbol{\Phi}\boldsymbol{w}$. Taking the gradient with respect to $\boldsymbol{w}$ yields

$$\nabla_{\boldsymbol{w}}\mathcal{L}(\boldsymbol{w}) = \frac{1}{\sqrt{N}}\boldsymbol{\Phi}^\top\left(\frac{1}{\sqrt{N}}\boldsymbol{\Phi}\boldsymbol{w} - \boldsymbol{y}\right) + \gamma\boldsymbol{w} = \left(\frac{1}{N}\boldsymbol{\Phi}^\top\boldsymbol{\Phi} + \gamma\boldsymbol{I}\right)\boldsymbol{w} - \frac{1}{\sqrt{N}}\boldsymbol{\Phi}^\top\boldsymbol{y}. \tag{17}$$

Setting this gradient to zero and solving for $\boldsymbol{W}$ leads to the canonical ridge regression solution

$$\boldsymbol{w} = \sqrt{N}\left(\boldsymbol{\Phi}^\top\boldsymbol{\Phi} - N\gamma\boldsymbol{I}\right)^{-1}\boldsymbol{\Phi}^\top\boldsymbol{y}. \tag{18}$$

We follow Bishop [42] to kernelize the expression. Note that the optimal weights satisfy

$$\boldsymbol{w} = -\frac{1}{\gamma\sqrt{N}}\boldsymbol{\Phi}^\top\left(\frac{1}{\sqrt{N}}\boldsymbol{\Phi}\boldsymbol{w} - \boldsymbol{y}\right) = \boldsymbol{\Phi}^\top\boldsymbol{a}, \quad \text{with} \quad \boldsymbol{a} = -\frac{1}{\gamma\boldsymbol{N}}\left(\frac{1}{\sqrt{N}}\boldsymbol{\Phi}\boldsymbol{w} - \boldsymbol{y}\right). \tag{19}$$

That is, the optimal weights are a linear combination of the $\boldsymbol{f}(x_p)'s$ of the training set. Substituting this expression into the original loss function results in

$$\begin{aligned}\mathcal{L}(\boldsymbol{a}) &= \frac{1}{2N}\boldsymbol{a}^\top\boldsymbol{\Phi}\boldsymbol{\Phi}^\top\boldsymbol{\Phi}\boldsymbol{\Phi}^\top\boldsymbol{a} - \frac{1}{\sqrt{N}}\boldsymbol{a}^\top\boldsymbol{\Phi}\boldsymbol{\Phi}^\top\boldsymbol{y} + \frac{1}{2}\boldsymbol{y}^T\boldsymbol{y} + \frac{\gamma}{2}\boldsymbol{a}^T\boldsymbol{\Phi}\boldsymbol{\Phi}^\top\boldsymbol{a} \\ &= \frac{N}{2}\boldsymbol{a}^\top\boldsymbol{K}\boldsymbol{K}\boldsymbol{a} - \sqrt{N}\boldsymbol{a}^T\boldsymbol{K}\boldsymbol{y} + \frac{1}{2}\boldsymbol{y}^T\boldsymbol{y} + \frac{\gamma N}{2}\boldsymbol{a}^\top\boldsymbol{K}\boldsymbol{a},\end{aligned} \tag{20}$$

where we have defined the kernel matrix $\boldsymbol{K} = \frac{1}{N}\boldsymbol{\Phi}\boldsymbol{\Phi}^\top$ with elements

$$[\boldsymbol{K}]_{ij} = \frac{1}{N}\boldsymbol{r}(\boldsymbol{x}_i)^\top\boldsymbol{r}(\boldsymbol{x}_j) = \frac{1}{N}\sum_n r(\boldsymbol{x}_i;\Omega_n)r(\boldsymbol{x}_j;\Omega_n) \approx \langle r(x_i;\Omega_n)r(x_j;\Omega_n)\rangle\,. \tag{21}$$

Taking the gradient of this loss with respect to $\boldsymbol{a}$ and setting it to zero results in

$$\boldsymbol{a} = \frac{1}{\sqrt{N}}\left(\boldsymbol{K} + \gamma\boldsymbol{I}\right)^{-1}\boldsymbol{y}. \tag{22}$$

Using $\boldsymbol{w} = \boldsymbol{\Phi}^\top\boldsymbol{a}$ and substituting the resulting expression back into $\hat{y}(\boldsymbol{x})$ finally yields

$$\hat{y}(\boldsymbol{x}) = \frac{1}{N}\boldsymbol{y}^T\left(\boldsymbol{K} + \gamma\boldsymbol{I}\right)^{-1}\boldsymbol{\Phi}\boldsymbol{r}(\boldsymbol{x}) = \boldsymbol{k}(\boldsymbol{x})^\top\left(\boldsymbol{K} + \gamma\boldsymbol{I}\right)^{-1}\boldsymbol{y}, \tag{23}$$

where we defined the p-th element of $\boldsymbol{k}(x)$ to be given by

$$[\boldsymbol{k}(\boldsymbol{x})]_p = \frac{1}{N}\boldsymbol{r}(\boldsymbol{x}_p)^\top\boldsymbol{r}(\boldsymbol{x}) \approx \langle\boldsymbol{r}(\boldsymbol{x}_p)\boldsymbol{r}(\boldsymbol{x})\rangle\,. \tag{24}$$

C.2 GP posterior solution

To find the GP posterior, we will assume observation noise σ^2, such that

$$y|\boldsymbol{x}, \hat{y} \sim \mathcal{N}\left(\hat{y}(\boldsymbol{x}), \sigma^2\right), \quad \text{and,} \quad \boldsymbol{y}|\boldsymbol{x}_{1:P}, \hat{\boldsymbol{y}} \sim \mathcal{N}\left(\boldsymbol{y}|\hat{\boldsymbol{y}}, \sigma^2\boldsymbol{I}\right). \tag{25}$$

Furthermore, by the GP prior we have

$$\hat{\boldsymbol{y}}|\boldsymbol{x}_{1:P} \sim \mathcal{N}\left(\boldsymbol{0}, \sigma_w^2\boldsymbol{K}\right). \tag{26}$$

Marginalizing over the GP thus yields

$$\boldsymbol{y}|\boldsymbol{x}_{1:P} \sim \mathcal{N}\left(\boldsymbol{0}, \sigma_w^2\boldsymbol{K} + \sigma^2\boldsymbol{I}\right). \tag{27}$$

To find the posterior $\hat{y}(\boldsymbol{x})|\boldsymbol{y}, \boldsymbol{x}_{1:P}$, we augment $\hat{\boldsymbol{y}}$ by $\hat{y}(\boldsymbol{x})$ to get $\tilde{\boldsymbol{y}} = (\hat{y}(\boldsymbol{x}_1), \ldots, \hat{y}(\boldsymbol{x}_P), \hat{y}(\boldsymbol{x}))^\top$, which is by the GP prior distributed as

$$\tilde{\boldsymbol{y}}|\boldsymbol{x}_{1:P}, x \sim \mathcal{N}\left(\boldsymbol{0}, \sigma_w^2\begin{bmatrix}\boldsymbol{K} & \boldsymbol{k}(\boldsymbol{x}) \\ \boldsymbol{k}(\boldsymbol{x})^\top & \langle f_n(\boldsymbol{x})^2\rangle\end{bmatrix}\right) \tag{28}$$

Using Gaussian identities, this results in the posterior mean and variance to be given by

$$\langle \hat{y}(\boldsymbol{x}) | \boldsymbol{x}_{1:P}, \boldsymbol{y} \rangle = \sigma_w^2 \boldsymbol{k}(\boldsymbol{x})^\top \left(\sigma_w^2 \boldsymbol{K} + \sigma^2 \boldsymbol{I} \right)^{-1} \boldsymbol{y} = \boldsymbol{k}(\boldsymbol{x})^\top \left(\boldsymbol{K} + \frac{\sigma^2}{\sigma_w^2} \boldsymbol{I} \right)^{-1} \boldsymbol{y}, \tag{29}$$

$$\begin{aligned} \text{Var}\left(\hat{y}(\boldsymbol{x}) | \boldsymbol{x}_{1:P}, \boldsymbol{y} \right) &= \sigma_w^2 \left\langle r(\boldsymbol{x}; \Omega_n)^2 \right\rangle - \sigma_w^2 \boldsymbol{k}(\boldsymbol{x})^\top \left(\sigma_w^2 \boldsymbol{K} + \sigma^2 \boldsymbol{I} \right)^{-1} \boldsymbol{k}(\boldsymbol{x}) \sigma_w^2 \\ &= \sigma_w^2 \left(\left\langle r(\boldsymbol{x}; \Omega_n)^2 \right\rangle - \boldsymbol{k}(\boldsymbol{x})^\top \left(\boldsymbol{K} + \frac{\sigma^2}{\sigma_w^2} \boldsymbol{I} \right)^{-1} \boldsymbol{k}(\boldsymbol{x}) \right). \end{aligned} \tag{30}$$

Together, these result in the posterior distribution

$$\hat{y}(\boldsymbol{x}) | \boldsymbol{y}, x_{1:P} \sim \mathcal{N}\left(\boldsymbol{k}(\boldsymbol{x})^\top \left(\boldsymbol{K} + \frac{\sigma^2}{\sigma_w^2} \boldsymbol{I} \right)^{-1} \boldsymbol{y}, \sigma_w^2 \left(\left\langle r(\boldsymbol{x}; \Omega_n)^2 \right\rangle - \boldsymbol{k}(\boldsymbol{x})^\top \left(\boldsymbol{K} + \frac{\sigma^2}{\sigma_w^2} \boldsymbol{I} \right)^{-1} \boldsymbol{k}(\boldsymbol{x}) \right) \right) \tag{31}$$

Comparing the mean to the kernel regression solution shows that

$$\gamma = \frac{\sigma^2}{\sigma_w^2}. \tag{32}$$

That is, the regularization strength is controlled by the fraction of the observation noise over the weight noise. This implies that, for a fixed λ, there exists a whole family of GPs that lead to the same posterior mean.

D Tuning distributions control the population kernel

The expression for the induced population kernel is given entry-wise as

$$K(\boldsymbol{x}_i, \boldsymbol{x}_j) = \langle r(\boldsymbol{x}_i; \Omega_n) r(\boldsymbol{x}_j; \Omega_n) \rangle_{p(\Omega)} = \int p(\Omega_n) r(\boldsymbol{x}_i; \Omega_n) r(\boldsymbol{x}_j; \Omega_n) d\Omega. \tag{33}$$

Because all tuning parameters are independently drawn (i.e., $p(\Omega) = \prod_p p(\omega_p)$, where each $p(\omega_p)$ is the probability density of some tuning property of the neurons (such as the centers)), we can find the kernels corresponding to distributional assumptions of tuning by marginalizing the above equation separately for each tuning distribution. In the following derivations, we assume that $\boldsymbol{x} \in \mathbb{R}^D$, unless otherwise specified.

D.1 A tuning mixture distribution gives rise to a weighted mixture of kernels

Suppose that one of the $p(\omega_p)$ can be written as

$$p(\omega_p) = \sum_{q=1}^{Q} w_q p_q(\omega_p) \tag{34}$$

where $p_q(\omega_p)$ is a proper density and $\sum_q w_q = 1$. As an example, we might write the tuning density as a weighted mixture of a uniform distribution and a normal distribution (as we consider in the main text), $p(\mu) = \lambda \mathcal{N}(\mu_R, \sigma_R^2) + (1 - \lambda)\mathcal{U}(-L, L); \lambda \in [0, 1]$. Then,

$$\begin{aligned} K(\boldsymbol{x}_i, \boldsymbol{x}_j; \{\omega_a\}_{a \neq p}) &= \int p(\omega_p) r(\boldsymbol{x}_i; \Omega_n) r(\boldsymbol{x}_j; \Omega_n) d\omega_p = \int \sum_{q=1}^{Q} w_a p_q(\omega_p) r(\boldsymbol{x}_i; \Omega_n) r(\boldsymbol{x}_j; \Omega_n) d\omega_p \\ &= \sum_{q=1}^{Q} w_q \int p_q(\omega_p) r(\boldsymbol{x}_i; \Omega_n) r(\boldsymbol{x}_j; \Omega_n) d\omega_p \\ &= \sum_{q=1}^{Q} w_q \left\langle K(\boldsymbol{x}_i, \boldsymbol{x}_j) \right\rangle_{p_q(\omega_q)}. \end{aligned} \tag{35}$$

In the case of $p(\mu) = \lambda\mathcal{N}(\mu_R, \sigma_R^2) + (1-\lambda)\mathcal{U}(-L, L); \lambda \in [0,1]$ mentioned above, this means that the kernel is a convex combination of the kernels induced by the two constituent tuning center distributions:

$$K(\boldsymbol{x}_i, \boldsymbol{x}_j) = \lambda \left\langle K(\boldsymbol{x}_i, \boldsymbol{x}_j) \right\rangle_{\mu \sim \mathcal{N}(\mu_R, \sigma_R^2)} + (1-\lambda) \left\langle K(\boldsymbol{x}_i, \boldsymbol{x}_j) \right\rangle_{\mu \sim \mathcal{U}(-L,L)} \tag{36}$$

We verify this analytical finding in simulated populations in Fig. S5.

D.2 A population with homogeneous widths and Gaussian tuning curves gives rise to a homogeneous RBF kernel

A homogeneous RBF kernel arises when $r(\boldsymbol{x}; \Omega_n) = \sqrt{a_n}\exp(-\|\boldsymbol{x} - \boldsymbol{\mu}_n\|^2/2\sigma_n^2)$, $a_n \sim p(a)$ where $\langle a_n \rangle = g$, $p(\sigma) = \delta(\sigma - \ell)$, and $p(\mu_{n,i}) = \mathcal{U}(-L, L)$ where $L \gg x_{\max}$, the maximum value of the stimulus $\boldsymbol{x}$. Indeed,

$$\begin{aligned} K(\boldsymbol{x}_i, \boldsymbol{x}_j; \{\boldsymbol{\mu}_n\}_n, \{\sigma_n\}_n) &= \int p(a_n) a_n \exp\left(-\frac{\|\boldsymbol{x}_i - \boldsymbol{\mu}_n\|^2}{2\sigma_n^2}\right) \exp\left(-\frac{\|\boldsymbol{x}_j - \boldsymbol{\mu}_n\|^2}{2\sigma_n^2}\right) da \\ &= g \exp\left(-\frac{\|\boldsymbol{x}_i - \boldsymbol{\mu}_n\|^2}{2\sigma_n^2} - \frac{\|\boldsymbol{x}_j - \boldsymbol{\mu}_n\|^2}{2\sigma_n^2}\right). \end{aligned} \tag{37}$$

Then

$$\begin{aligned} K(\boldsymbol{x}_i, \boldsymbol{x}_j; \{\sigma_n\}_n) &= \eta \int p(\boldsymbol{\mu}_n) g \exp\left(-\frac{\|\boldsymbol{x}_i - \boldsymbol{\mu}_n\|^2}{2\sigma_n^2} - \frac{\|\boldsymbol{x}_j - \boldsymbol{\mu}_n\|^2}{2\sigma_n^2}\right) d\boldsymbol{\mu} \\ &= \frac{\eta}{(2L)^D} g \pi^{D/2} \sigma_n^D \exp\left(-\frac{\|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2}{4\sigma_n^2}\right). \end{aligned} \tag{38}$$

where η reflects normalization for the tuning density. Here and elsewhere in the paper, we let $\eta = (2L)^D$ to compensate for the width of the tuning center distribution with respect to the stimulus distribution, such that

$$K(\boldsymbol{x}_i, \boldsymbol{x}_j; \{\sigma_n\}_n) = g \pi^{D/2} \sigma_n^D \exp\left(-\frac{\|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2}{4\sigma_n^2}\right). \tag{39}$$

And finally,

$$K(\boldsymbol{x}_i, \boldsymbol{x}_j) = \int p(\sigma_n) g \pi^{D/2} \sigma_n^D \exp\left(-\frac{\|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2}{4\sigma_n^2}\right) d\sigma_n = g \pi^{D/2} \ell^D \exp\left(-\frac{\|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2}{4\ell^2}\right). \tag{40}$$

D.3 A population with heterogeneous widths and Gaussian tuning curves gives rise scale-mixture RBF kernel and the rational quadratic kernel

If we instead assume $\sigma_n \sim p(\sigma)$ that is not a delta function, but leave all other assumptions from the RBF kernel example above the same, we obtain more generally that that population induces a scale-mixture kernel:

$$K(\boldsymbol{x}_i, \boldsymbol{x}_j) = \int p(\sigma) K^{\text{RBF}}(\boldsymbol{x}_i, \boldsymbol{x}_j; \sigma) d\sigma \tag{41}$$

where $K^{\text{RBF}}(\boldsymbol{x}_i, \boldsymbol{x}_j; s) = g\pi^{D/2} s^D \exp(-\|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2/4s^2)$. This is the most general form of an isotropic Gaussian kernel [32]. Interesting special cases arise for particular distributions of the length scale. In particular, setting $p(\sigma^2) = \text{InvGamma}(\alpha, \alpha\ell^2)$ yields the rational quadratic (RQ) kernel:

$$K^{\text{RQ}}(\boldsymbol{x}_i, \boldsymbol{x}_j) = \int \text{InvGamma}(\sigma^2; \alpha, \alpha\ell^2) K^{\text{RBF}}(\boldsymbol{x}_i, \boldsymbol{x}_j; \sigma) d\sigma \propto \left(1 + \frac{\|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2}{2\alpha_{\text{eff}}\ell_{\text{eff}}^2}\right)^{-\alpha_{\text{eff}}} \tag{42}$$

where $\alpha_{\text{eff}} = \alpha - D/2$ (where $\alpha > D/2$), and $\ell_{\text{eff}} = \frac{2\alpha}{\alpha - D/2}\ell^2$, and the exact expression for this kernel is given by

$$K^{\text{RQ}}(\boldsymbol{x}_i, \boldsymbol{x}_j) = \pi^{D/2} \frac{\Gamma(\alpha - D/2)}{\Gamma(\alpha)} (\alpha\ell^2)^{D/2} \left(1 + \frac{\|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2}{4\alpha\ell^2}\right)^{D/2-\alpha}. \tag{43}$$

D.4 A population with a mixture distribution of uniform and Gaussian distributed centers and Gaussian tuning curves gives rise to a weighted mixture of an RBF kernel and a modulated RBF kernel

In this exampled, we restrict ourselves to the case of one-dimensional x's. For simplicity, we assume that the population exhibits a shared, homogeneous width ℓ across all neurons in the population. All neurons similarly share the same gain, $\sqrt{g}$. The only source of heterogeneity in this case comes from the distribution of tuning centers, which we assume is $p(\mu) = \lambda\mathcal{N}(\mu_R, \sigma_R^2) + (1-\lambda)\mathcal{U}(-L, L); \lambda \in [0, 1]$. As we showed above, the kernel for this population is given by $K(x_i, x_j) = \lambda \left\langle K(x_i, x_j) \right\rangle_{\mu\sim\mathcal{N}(\mu_R,\sigma_R^2)} + (1-\lambda) \left\langle K(x_i, x_j) \right\rangle_{\mu\sim\mathcal{U}(-L,L)}$. The latter component is a standard RBF kernel with bandwidth ℓ. The first component is a "modulated RBF kernel" [32], and is given by

$$\begin{aligned} K^{\text{mRBF}}(x_i, x_j) &= \frac{g}{\sigma_R\sqrt{2\pi}} \int \exp\left(\frac{-(x_i-\mu_n)^2}{2\ell^2} - \frac{(x_j-\mu_n)^2}{2\ell^2} - \frac{(\mu_n-\mu_R)(\mu_n-\mu_R)}{2\sigma_R^2}\right) d\mu_n \\ &= \frac{1}{\sigma_R}\left(\frac{2}{\ell^2} + \frac{1}{\sigma_R^2}\right)^{-1/2} \exp\left(-\frac{(x_i-\mu_R)^2}{2(2\sigma_R^2+\ell^2)}\right) \exp\left(-\frac{(x_i-x_j)^2}{2(2\ell^2+\ell^4/\sigma_R^2)}\right) \\ &\quad \times \exp\left(-\frac{(x_j-\mu_R)^2}{2(2\sigma_R^2+\ell^2)}\right) \end{aligned} \tag{44}$$

D.5 A population with multimodal Gaussian tuning curves gives rise to a scaled and shifted RBF kernel

Many neural populations exhibit multimodal tuning curves. Here, we explore what kernels emerge in such populations. We restrict ourselves to the study of tuning curves that can be written as

$$r(\boldsymbol{x}; \Omega_n) = \sum_{p=1}^{P} w_{n,p} a_{n,p} \exp\left(\frac{-\|\boldsymbol{x} - \boldsymbol{\mu}_{n,p}\|^2}{2\sigma_{n,p}^2}\right), \tag{45}$$

where $w_{n,p} \sim p(w)$, $a_{n,p} \sim p(a)$, $\sigma_{n,p} \sim p(\sigma)$ and $\boldsymbol{\mu}_{n,p} \sim p(\boldsymbol{\mu})$. In other words, each neuron's tuning curve can be written as a weighted sum of P Gaussian subfields, where all tuning curve parameters across neurons and subfields are sampled i.i.d.

Then,

$$\begin{aligned} K^{\text{MM}}(\boldsymbol{x}_i, \boldsymbol{x}_j) &= \left\langle \left(\sum_{p=1}^{P} w_{n,p} a_{n,p} \exp\left(\frac{-\|\boldsymbol{x}_i - \boldsymbol{\mu}_{n,p}\|^2}{2\sigma_{n,p}^2}\right)\right) \right. \\ &\qquad \left. \times \left(\sum_{p=1}^{P} w_{n,p} a_{n,p} \exp\left(\frac{-\|\boldsymbol{x}_j - \boldsymbol{\mu}_{n,p}\|^2}{2\sigma_{n,p}^2}\right)\right) \right\rangle_{p(\Omega)} \\ &= \sum_{p,l} \left\langle w_{n,p} w_{n,l} a_{n,p} a_{n,l} \exp\left(\frac{-\|\boldsymbol{x}_i - \boldsymbol{\mu}_{n,p}\|^2}{2\sigma_{n,p}^2} - \frac{\|\boldsymbol{x}_j - \boldsymbol{\mu}_{n,l}\|^2}{2\sigma_{n,l}^2}\right) \right\rangle_{p(\Omega)} \\ &= \sum_{p=1}^{P} \left\langle w_{n,p}^2 \right\rangle_{p(w)} \left\langle a_{n,p}^2 \right\rangle_{p(a)} \left\langle \exp\left(\frac{-\|\boldsymbol{x}_i - \boldsymbol{\mu}_{n,p}\|^2}{2\sigma_{n,p}^2} - \frac{\|\boldsymbol{x}_j - \boldsymbol{\mu}_{n,p}\|^2}{2\sigma_{n,p}^2}\right) \right\rangle_{p(\sigma,\boldsymbol{\mu})} \\ &\quad + \sum_{p=1}^{P} \sum_{l\neq p} \left\langle w_{n,p} \right\rangle_{p(w)}^2 \left\langle a_{n,p} \right\rangle_{p(a)}^2 \left\langle \exp\left(\frac{-\|\boldsymbol{x}_i - \boldsymbol{\mu}_{n,p}\|^2}{2\sigma_{n,p}^2} - \frac{\|\boldsymbol{x}_j - \boldsymbol{\mu}_{n,l}\|^2}{2\sigma_{n,l}^2}\right) \right\rangle_{p(\sigma,\boldsymbol{\mu})} \end{aligned} \tag{46}$$

Then, in the first term in the last line above, marginalizing over the tuning center distribution, $p(\boldsymbol{\mu})$, yields the RBF kernel from Eq. (38). In the second term, because the tuning centers are independent, the marginalization factorizes over the independent tuning centers, so that marginalizing over the tuning center distribution simply yields the squared normalization constant for a Gaussian distribution.

$$\begin{aligned} K^{\text{MM}}(\boldsymbol{x}_i, \boldsymbol{x}_j) &= P \left\langle w_{n,p}^2 \right\rangle_{p(w)} \left\langle a_{n,p}^2 \right\rangle_{p(a)} \left\langle K^{\text{RBF}}(\boldsymbol{x}_i, \boldsymbol{x}_j) \right\rangle_{p(\sigma)} \\ &\quad + P(P-1) \left\langle w_{n,p} \right\rangle_{p(w)}^2 \left\langle a_{n,p} \right\rangle_{p(a)}^2 \pi^D \left\langle \sigma_{n,p}^D \right\rangle_{p(\sigma)}^2 . \end{aligned} \tag{47}$$

where the explicit sums are replaced by factors P or $P(P-1)$, respectively, because all parameters are identically distributed. As in previous sections, if $p(\sigma) = \delta(\sigma - \ell)$, then

$$\begin{aligned} K^{\text{MM}}(\boldsymbol{x}_i, \boldsymbol{x}_j) = P \left\langle w_{n,p}^2 \right\rangle_{p(w)} \left\langle a_{n,p}^2 \right\rangle_{p(a)} K^{\text{RBF}}(\boldsymbol{x}_i, \boldsymbol{x}_j; \ell) \\ + P(P-1) \left\langle w_{n,p} \right\rangle_{p(w)}^2 \left\langle a_{n,p} \right\rangle_{p(a)}^2 \pi^D \ell^{2D}, \end{aligned} \tag{48}$$

or else that the multimodal tuning curve kernel is the standard RBF kernel scaled by the number of subfields, P and a constant kernel. If instead we assume a mixing distribution on $p(\sigma)$, the multimodal kernel becomes the sum of a scale-mixture RBF kernel and a constant kernel whose magnitude is proportional to the D-th moments of the mixing distribution.

D.6 A population with periodic tuning curves give rise to a periodic kernel

Some neurons, such as grid cells, exhibit periodic tuning. Here, we show that a population with periodic tuning curves gives rise to periodic kernel. This example is presented in [32], and we re-state it here for completeness. Here, we assume that inputs, x, are one-dimensional. We assume a population of neurons that are tuned to x according to

$$r(x; \Omega_n) = \sqrt{a_n} \exp\left(\frac{-\|\boldsymbol{u}(x) - \boldsymbol{\mu}_n\|^2}{2\sigma_n^2} \right) \tag{49}$$

where $\boldsymbol{u}(x) = (\cos(x), \sin(x))^\top$ and the tuning centers are distributed uniformly in $\boldsymbol{u}$-space. Then, as before, the kernel emerges from marginalizing over the respective tuning parameters:

$$\begin{aligned} K^P(x_i, x_j) &= \langle a_n \rangle_{p(a)} \left\langle \exp\left(-\frac{\|\boldsymbol{u}(x_i) - \boldsymbol{\mu}_n\|^2}{2\sigma_n^2} - \frac{\|\boldsymbol{u}(x_j) - \boldsymbol{\mu}_n\|^2}{2\sigma_n^2} \right) \right\rangle_{p(\boldsymbol{\mu}, \sigma)} \\ &= g\sqrt{\pi}\sigma_n \exp\left(\frac{-\|\boldsymbol{u}(x_i) - \boldsymbol{u}(x_j)\|^2}{4\sigma_n^2} \right) \\ &= g\sqrt{\pi}\sigma_n \exp\left(\frac{-\sin^2\left(\frac{x_i - x_j}{2}\right)}{\sigma_n^2} \right), \end{aligned} \tag{50}$$

where the last line follows from the identity $(\cos(x) - \cos(x'))^2 + (\sin(x) - \sin(x'))^2 = 4\sin^2((x - x')/2)$ [32]. Then, as before, if $p(\sigma) = \delta(\sigma - \ell)$, then the population's kernel is simply the periodic kernel above with length-scale ℓ:

$$K^P(x_i, x_j; \ell) = g\sqrt{\pi}\ell \exp\left(\frac{-\sin^2\left(\frac{x_i - x_j}{2}\right)}{\ell^2} \right). \tag{51}$$

Alternatively, when $p(\sigma)$ exhibits variability, we obtain a scale-mixture periodic kernel controlled by the mixing distribution $p(\sigma)$. As a special case, let $\tau = 1/\sigma_n^2$ and assume $\tau \sim \Gamma(\alpha, \beta)$. Then we obtain

$$K^P(x_i, x_j) = \frac{g\sqrt{\pi}\,\Gamma(\alpha - \frac{1}{2})}{\Gamma(\alpha)\sqrt{\beta}} \left(1 - \beta \sin^2\left(\frac{x_i - x_j}{2} \right) \right)^{-(\alpha - \frac{1}{2})} \tag{52}$$

which mirrors the form of the standard rational quadratic kernel above.

E The generalization error for mismatched student and teacher GPs

Following Sollich [34, 43], we consider the student-teacher GP learning scenario. In this scenario, the teacher draws a target function $y^* \sim \mathcal{GP}(0, K_T)$, uniformly draws $\boldsymbol{x} \sim \mathcal{U}[x_{\min}, x_{\max}]^D$ and then obtains a dataset $\mathcal{D} = \{\boldsymbol{x}_i, y_i\}_{i=1}^P$ where $\boldsymbol{y}_i \sim \mathcal{N}(y^*(\boldsymbol{x}_i), \sigma_T^2)$. First we show how to compute the Bayes error.

E.1 The Bayes error

The teacher's posterior over a test point $\boldsymbol{x}$ conditioned on its dataset, $p(y^*|\mathcal{D})$, is Gaussian with mean and variance

$$\begin{aligned} \langle y^*(\boldsymbol{x})\rangle_{p(y^*|\mathcal{D})} &= \boldsymbol{k}_T(\boldsymbol{x})^\top (\boldsymbol{K}_T + \sigma_T^2 \boldsymbol{I})^{-1} \boldsymbol{y} \\ \left\langle \left(y^*(\boldsymbol{x}) - \langle y^*(\boldsymbol{x})\rangle_{p(y^*|\mathcal{D})} \right)^2 \right\rangle_{p(y^*|\mathcal{D})} &= K_T(\boldsymbol{x},\boldsymbol{x}) - \boldsymbol{k}_T(\boldsymbol{x})^\top (\boldsymbol{K}_T + \sigma_T^2 \boldsymbol{I})^{-1} \boldsymbol{k}_T(\boldsymbol{x}) \end{aligned} \tag{53}$$

where $\boldsymbol{k}_T(\boldsymbol{x})$ is the column of the teacher kernel corresponding to the point $\boldsymbol{x}$. Note that this expression gives the generalization error at $\boldsymbol{x}$ for the teacher: Note that the second expression above is just the mean (w.r.t the teacher posterior) squared deviation between the teacher output and its posterior mean prediction conditioned on the data. In other words, the generalization error for the teacher on a test point $\boldsymbol{x}$ is simply $K(\boldsymbol{x},\boldsymbol{x}) - \boldsymbol{k}_T(\boldsymbol{x})^\top (\boldsymbol{K}_T + \sigma_T^2 \boldsymbol{I})^{-1} \boldsymbol{k}_T(\boldsymbol{x})$. Averaging this over $p(\boldsymbol{x})$ and $p(\mathcal{D})$ yields the full generalization error for the teacher:

$$\begin{aligned} \varepsilon_{\text{Bayes}}(P) &= \left\langle K_T(\boldsymbol{x},\boldsymbol{x}) - \boldsymbol{k}_T(\boldsymbol{x})^\top (\boldsymbol{K}_T + \sigma_T^2 \boldsymbol{I})^{-1} \boldsymbol{k}_T(\boldsymbol{x}) \right\rangle_{p(\boldsymbol{x},\mathcal{D})} \\ &= \left\langle K_T(\boldsymbol{x},\boldsymbol{x}) + \text{Tr}\left(\left\langle \boldsymbol{k}_T(\boldsymbol{x}) \boldsymbol{k}_T(\boldsymbol{x})^\top \right\rangle_{p(\boldsymbol{x})} (\boldsymbol{K}_T + \sigma_T^2 \boldsymbol{I})^{-1} \right) \right\rangle_{p(\{\boldsymbol{x}_i\},\boldsymbol{x})} \end{aligned} \tag{54}$$

where, because the terms in this expression do not depend on the associated $\boldsymbol{y}_i's$ we only average the dataset distribution over the constituent $\boldsymbol{x}_i's$.

We can make further headway in this expression by exploiting Mercer's theorem, which admits an eigendecomposition of the teacher covariance matrix into an infinite basis:

$$K_T(\boldsymbol{x},\boldsymbol{x}') = \sum_{i=1}^{\infty} \lambda_i \psi_i(\boldsymbol{x}) \psi_i(\boldsymbol{x}') \tag{55}$$

where λ_i is the i-th eigenvalue and $\psi_i(\cdot)$ is the i-th eigenfunction. Let $\boldsymbol{\Lambda}$ be the diagonal matrix of eigenvalues λ_i, and let $\boldsymbol{\Psi}$ be a matrix whose columns are the eigenfunctions. Then $\boldsymbol{K}_T = \boldsymbol{\Psi}\boldsymbol{\Lambda}\boldsymbol{\Psi}^\top$. Such a representation allows us to simplify the expression for $\varepsilon_{\text{Bayes}}(P)$ considerably (see [34, 43] for more details):

$$\varepsilon_{\text{Bayes}}(P) = \left\langle \text{Tr}(\boldsymbol{\Lambda}) + \text{Tr}\left((\sigma_T^2 \boldsymbol{I} + \boldsymbol{\Psi}\boldsymbol{\Lambda}\boldsymbol{\Psi}^\top)^{-1} \boldsymbol{\Psi}\boldsymbol{\Lambda}^2\boldsymbol{\Psi}^\top \right) \right\rangle_{p(\{\boldsymbol{x}_i\})}, \tag{56}$$

which can then be further simplified to

$$\varepsilon_{\text{Bayes}}(P) = \left\langle \sum_j \lambda_j^{T,\{\boldsymbol{x}_i\}} \right\rangle_{p(\{\boldsymbol{x}_i\})} \tag{57}$$

where $\lambda_j^{T,\{\boldsymbol{x}_i\}}$ are the eigenvalues of the data-dependent matrix $\left(\boldsymbol{\Lambda}^{-1} + \sigma_T^{-2}\boldsymbol{\Psi}^\top\boldsymbol{\Psi}\right)^{-1}$. The data-dependence comes through the $\boldsymbol{\Psi}$ matrices.

E.2 The student's error

The student computes a posterior over the teacher's dataset, which is a function of its own kernel, K_S and its own assumption on the noise model, $y_i \sim \mathcal{N}(y^*(\boldsymbol{x}_i), \sigma_S^2)$ where y^* is the true function. Our goal is to understand how the generalization error of the student varies with its kernel, and, in particular, how this variation depends on the degree of mismatch between the student and teacher kernels.

First, it is clear that if $K_S = K_T$ and $\sigma_T^2 = \sigma_S^2$, then the student will achieve the Bayes error on the learning problem. This is because the student and teacher posteriors over new test points $\boldsymbol{x}$ will be identical. But what happens if $K_S \neq K_T$ (and $\sigma_T \neq \sigma_S$)? We return to the expression for the generalization error from above, $\left\langle \left(y^*(\boldsymbol{x}) - \langle y^*(\boldsymbol{x})\rangle_{p(y^*|\mathcal{D})} \right)^2 \right\rangle_{p(y^*|\mathcal{D})}$. Remember that this is the generalization error from the perspective of the teacher. The student obtains its own posterior, which is Gaussian, whose test point, $\boldsymbol{x}$, mean is given by

$$\langle \hat{y}_S(\boldsymbol{x})\rangle_{p(h|\mathcal{D})} = \boldsymbol{k}_S(\boldsymbol{x})^\top \left(\boldsymbol{K}_S + \sigma_S^2 \boldsymbol{I} \right)^{-1} \boldsymbol{y}. \tag{58}$$

Then the mean squared deviation w.r.t the teacher posterior at a test point $\boldsymbol{x}$ is $\left\langle \left(y^*(\boldsymbol{x}) - \langle \hat{y}_S(\boldsymbol{x}) \rangle_{p(\hat{y}_S|\mathcal{D})} \right)^2 \right\rangle_{p(y^*|\mathcal{D})}$, where all we have done is replace the teacher's mean prediction with the student's. Then,

$$\varepsilon(P) = \left\langle \left\langle \left(y^*(\boldsymbol{x}) - \langle \hat{y}_S(\boldsymbol{x}) \rangle_{p(\hat{y}_S|\mathcal{D})} \right)^2 \right\rangle_{p(y^*|\mathcal{D})} \right\rangle_{p(\boldsymbol{x},\mathcal{D})}. \tag{59}$$

We will next decompose this equation into two terms. One will be the teacher's irreducible Bayes error, and the other will be the additional error incurred through misalignment of the student and teacher. The misalignment term has a straight-forward interpretation, as the mean squared difference between the posterior means of the student and teacher. Indeed,

$$\begin{aligned}
\varepsilon(P) &= \left\langle \left\langle \left(y^*(\boldsymbol{x}) - \langle \hat{y}_S(\boldsymbol{x}) \rangle_{p(\hat{y}_S|\mathcal{D})} \right)^2 \right\rangle_{p(y^*|\mathcal{D})} \right\rangle_{p(\boldsymbol{x},\mathcal{D})} \\
&= \left\langle \left\langle \left(y^*(\boldsymbol{x}) - \langle \hat{y}_S(\boldsymbol{x}) \rangle_{p(\hat{y}_S|\mathcal{D})} + \langle y^*(\boldsymbol{x}) \rangle_{p(y^*|\mathcal{D})} - \langle y^*(\boldsymbol{x}) \rangle_{p(y^*|\mathcal{D})} \right)^2 \right\rangle_{p(y^*|\mathcal{D})} \right\rangle_{p(\boldsymbol{x},\mathcal{D})} \\
&= \left\langle \left\langle \left(\left(y^*(\boldsymbol{x}) - \langle y^*(\boldsymbol{x}) \rangle_{p(y^*|\mathcal{D})} \right) + \left(\langle y^*(\boldsymbol{x}) \rangle_{p(y^*|\mathcal{D})} - \langle \hat{y}_S(\boldsymbol{x}) \rangle_{p(\hat{y}_S|\mathcal{D})} \right) \right)^2 \right\rangle_{p(y^*|\mathcal{D})} \right\rangle_{p(\boldsymbol{x},\mathcal{D})} \\
&= \left\langle \left\langle \left(y^*(\boldsymbol{x}) - \langle y^*(\boldsymbol{x}) \rangle_{p(y^*|\mathcal{D})} \right)^2 - 2 \left(y^*(\boldsymbol{x}) - \langle y^*(\boldsymbol{x}) \rangle_{p(y^*|\mathcal{D})} \right) \left(\langle y^*(\boldsymbol{x}) \rangle_{p(y^*|\mathcal{D})} - \langle \hat{y}_S(\boldsymbol{x}) \rangle_{p(\hat{y}_S|\mathcal{D})} \right) + \right. \right. \\
&\qquad \left. \left. \left(\langle y^*(\boldsymbol{x}) \rangle_{p(y^*|\mathcal{D})} - \langle \hat{y}_S(\boldsymbol{x}) \rangle_{p(\hat{y}_S|\mathcal{D})} \right)^2 \right\rangle_{p(y^*|\mathcal{D})} \right\rangle_{p(\boldsymbol{x},\mathcal{D})}
\end{aligned} \tag{60}$$

This expression simplifies considerably. Pushing in the expectation over the teacher posterior, we see that the first term is simply

$$\left\langle \left\langle \left(y^*(\boldsymbol{x}) - \langle y^*(\boldsymbol{x}) \rangle_{p(y^*|\mathcal{D})} \right)^2 \right\rangle_{p(y^*|\mathcal{D})} \right\rangle_{p(\boldsymbol{x},\mathcal{D})} = \varepsilon_{\text{Bayes}}(P), \tag{61}$$

the second term vanishes,

$$\left\langle \left\langle 2 \left(y^*(\boldsymbol{x}) - \langle y^*(\boldsymbol{x}) \rangle_{p(y^*|\mathcal{D})} \right) \right\rangle_{p(y^*|\mathcal{D})} \left(\langle y^*(\boldsymbol{x}) \rangle_{p(y^*|\mathcal{D})} - \langle \hat{y}_S(\boldsymbol{x}) \rangle_{p(\hat{y}_S|\mathcal{D})} \right) \right\rangle_{p(\boldsymbol{x},\mathcal{D})} = 0, \tag{62}$$

and the third term is independent of the teacher posterior

$$\left\langle \left\langle \left(\langle y^*(\boldsymbol{x}) \rangle_{p(y^*|\mathcal{D})} - \langle \hat{y}_S(\boldsymbol{x}) \rangle_{p(\hat{y}_S|\mathcal{D})} \right)^2 \right\rangle_{p(y^*|\mathcal{D})} \right\rangle_{p(\boldsymbol{x},\mathcal{D})} = \left\langle \left(\langle y^*(\boldsymbol{x}) \rangle_{p(y^*|\mathcal{D})} - \langle \hat{y}_S(\boldsymbol{x}) \rangle_{p(\hat{y}_S|\mathcal{D})} \right)^2 \right\rangle_{p(\boldsymbol{x},\mathcal{D})}. \tag{63}$$

This third term is the misalignment term, and is simply the average squared deviation between the student and teacher posterior means:

$$\varepsilon_{\text{Align}}(P) = \left\langle \left(\langle y^*(\boldsymbol{x}) \rangle_{p(y^*|\mathcal{D})} - \langle \hat{y}_S(\boldsymbol{x}) \rangle_{p(\hat{y}_S|\mathcal{D})} \right)^2 \right\rangle_{p(\boldsymbol{x},\mathcal{D})} \tag{64}$$

Before proceeding with computing this quantity, we note that is the average of a squared term, and so $\varepsilon_{\text{Align}}(P) \geq 0$. In other words, the student can, on average, only do as good as or worse than the teacher, but not better.

Because the student and teacher are both GPs with known posterior means, expanding $\varepsilon_{\text{Align}}(P)$ is straightforward:

$$\begin{aligned}
\varepsilon_{\text{Align}}(P) = \boldsymbol{k}_T(\boldsymbol{x})^\top (\boldsymbol{K}_T + \sigma_T^2 \boldsymbol{I})^{-1} \boldsymbol{y}\boldsymbol{y}^\top (\boldsymbol{K}_T + \sigma_T^2 \boldsymbol{I})^{-\top} \boldsymbol{k}_T(\boldsymbol{x}) - \\
2\boldsymbol{k}_T(\boldsymbol{x})^\top (\boldsymbol{K}_T + \sigma_T^2 \boldsymbol{I})^{-1} \boldsymbol{y}\boldsymbol{y}^\top \left(\boldsymbol{K}_S + \sigma_S^2 \boldsymbol{I} \right)^{-\top} \boldsymbol{k}_S(\boldsymbol{x}) + \\
\boldsymbol{k}_S(\boldsymbol{x})^\top \left(\boldsymbol{K}_S + \sigma_S^2 \boldsymbol{I} \right)^{-1} \boldsymbol{y}\boldsymbol{y}^\top \left(\boldsymbol{K}_S + \sigma_S^2 \boldsymbol{I} \right)^{-\top} \boldsymbol{k}_S(\boldsymbol{x}).
\end{aligned} \tag{65}$$

Note the presence of $\boldsymbol{y}\boldsymbol{y}^\top$ in several of the terms above. Note also that we plan to take an average over $p(\mathcal{D})$. We can rewrite this distribution as $p(\mathcal{D}) = p(\{y_i\}|\{\boldsymbol{x}_i\})p(\{\boldsymbol{x}_i\})$. Then note that first averaging over the conditioned outputs, $\left\langle \boldsymbol{y}\boldsymbol{y}^\top \right\rangle_{p(\{y_i\}|\{\boldsymbol{x}_i\})} = \boldsymbol{K}_T + \sigma_T^2 \boldsymbol{I}$. Applying that average above and also simultaneously averaging over $p(\boldsymbol{x})$, we get

$$\begin{aligned}\varepsilon_{\text{Align}}(P) = &\left\langle \text{Tr}\left(\left\langle \boldsymbol{k}_T(\boldsymbol{x})\boldsymbol{k}_T(\boldsymbol{x})^\top \right\rangle_{p(\boldsymbol{x})} \left(\boldsymbol{K}_T + \sigma_T^2\boldsymbol{I}\right)^{-1} \right) \right\rangle_{p(\{\boldsymbol{x}_i\})} \\ &- 2\left\langle \text{Tr}\left(\left\langle \boldsymbol{k}_S(\boldsymbol{x})\boldsymbol{k}_T(\boldsymbol{x})^\top \right\rangle_{p(\boldsymbol{x})} \left(\boldsymbol{K}_S + \sigma_S^2\boldsymbol{I}\right)^{-1} \right) \right\rangle_{p(\{\boldsymbol{x}_i\})} \\ &+ \left\langle \text{Tr}\left(\left\langle \boldsymbol{k}_S(\boldsymbol{x})\boldsymbol{k}_S(\boldsymbol{x})^\top \right\rangle_{p(\boldsymbol{x})} \left(\boldsymbol{K}_S + \sigma_S^2\boldsymbol{I}\right)^{-1} \left(\boldsymbol{K}_T + \sigma_T^2\boldsymbol{I}\right) \left(\boldsymbol{K}_S + \sigma_S^2\boldsymbol{I}\right)^{-1} \right) \right\rangle_{p(\{\boldsymbol{x}_i\})}. \end{aligned} \tag{66}$$

Right away, it is straight-forward to see that if $K_T = K_S$ and $\sigma_T = \sigma_S$, then $\varepsilon_{\text{Align}}(P) = 0$.

F The gap in the student and teacher evidence is the average KL divergence between the student and teacher marginal likelihoods

Let $\mathcal{L}_T = \log p(\{y_i\}|\{\boldsymbol{x}\}_i, \boldsymbol{\theta}_T)$ and $\mathcal{L}_S = \log p(\{y_i\}|\{\boldsymbol{x}\}_i, \boldsymbol{\theta}_S)$ be the teacher and student log evidences, respectively, for a dataset $\mathcal{D} = \{\boldsymbol{x}_i, y_i\}_{i=1}^P$. The parameters of each model are encapsulated in $\boldsymbol{\theta}_T, \boldsymbol{\theta}_S$, respectively. The teacher's model is composed of a prior that is a zero-mean GP with kernel K_T, $y^* \sim \mathcal{GP}(0, K_T)$ and a likelihood model $y_i \sim \mathcal{N}(y^*(\boldsymbol{x}_i), \sigma_T^2)$. The student's model is also composed of a GP prior with kernel K_S and Gaussian likelihood with variance σ_S^2. For a dataset $\mathcal{D}$ of size P, the log-evidences are

$$\mathcal{L}_T(\mathcal{D}) = -\frac{1}{2}\boldsymbol{y}^\top \left(\boldsymbol{K}_T + \sigma_T^2\boldsymbol{I}\right)^{-1}\boldsymbol{y} - \frac{1}{2}\log|\boldsymbol{K}_T + \sigma_T^2\boldsymbol{I}| - \frac{P}{2}\log(2\pi) \tag{67}$$

$$\mathcal{L}_S(\mathcal{D}) = -\frac{1}{2}\boldsymbol{y}^\top \left(\boldsymbol{K}_S + \sigma_S^2\boldsymbol{I}\right)^{-1}\boldsymbol{y} - \frac{1}{2}\log|\boldsymbol{K}_S + \sigma_S^2\boldsymbol{I}| - \frac{P}{2}\log(2\pi). \tag{68}$$

The evidence clearly fluctuates with the exact composition of $\mathcal{D}$. We thus characterize their relationship averaged over datasets, exploring their relationship on average as a function only of the dataset size and their respective models. Averaged over datasets $\mathcal{D} \sim p(\mathcal{D})$, these expressions become

$$\left\langle \mathcal{L}_T(\mathcal{D}) \right\rangle_{p(\mathcal{D})} = -\frac{1}{2}\text{Tr}\left(\left\langle \boldsymbol{y}\boldsymbol{y}^\top \left(\boldsymbol{K}_T + \sigma_T^2\boldsymbol{I}\right)^{-1} \right\rangle_{p(\mathcal{D})} \right) - \frac{1}{2}\left\langle \log|\boldsymbol{K}_T + \sigma_T^2\boldsymbol{I}| \right\rangle_{p(\mathcal{D})} - \frac{P}{2}\log(2\pi) \tag{69}$$

$$\left\langle \mathcal{L}_S(\mathcal{D}) \right\rangle_{p(\mathcal{D})} = -\frac{1}{2}\text{Tr}\left(\left\langle \boldsymbol{y}\boldsymbol{y}^\top \left(\boldsymbol{K}_S + \sigma_S^2\boldsymbol{I}\right)^{-1} \right\rangle_{p(\mathcal{D})} \right) - \frac{1}{2}\left\langle \log|\boldsymbol{K}_S + \sigma_S^2\boldsymbol{I}| \right\rangle_{p(\mathcal{D})} - \frac{P}{2}\log(2\pi). \tag{70}$$

Note that as above, if we partition $p(\mathcal{D}) = p(\{y_i\}|\{\boldsymbol{x}_i\})p(\{\boldsymbol{x}_i\})$ then $\left\langle \boldsymbol{y}\boldsymbol{y}^\top \right\rangle_{p(\{y_i\}|\{\boldsymbol{x}_i\})} = \boldsymbol{K}_T + \sigma_T^2\boldsymbol{I}$, and this is the only term in the above two equations that depends on the conditioned $\{y_i\}$, so

$$\begin{aligned}\left\langle \mathcal{L}_T(\mathcal{D}) \right\rangle_{p(\mathcal{D})} = -\frac{1}{2}\Bigg(&\text{Tr}\left(\left\langle \left(\boldsymbol{K}_T + \sigma_T^2\boldsymbol{I}\right)\left(\boldsymbol{K}_T + \sigma_T^2\boldsymbol{I}\right)^{-1} \right\rangle_{p(\{\boldsymbol{x}_i\})} \right) \\ &+ \left\langle \log|\boldsymbol{K}_T + \sigma_T^2\boldsymbol{I}| \right\rangle_{p(\{\boldsymbol{x}_i\})} + P\log(2\pi) \Bigg), \end{aligned} \tag{71}$$

$$\begin{aligned}\left\langle \mathcal{L}_S(\mathcal{D}) \right\rangle_{p(\mathcal{D})} = -\frac{1}{2}\Bigg(&\text{Tr}\left(\left\langle \left(\boldsymbol{K}_T + \sigma_T^2\boldsymbol{I}\right)\left(\boldsymbol{K}_S + \sigma_S^2\boldsymbol{I}\right)^{-1} \right\rangle_{p(\{\boldsymbol{x}_i\})} \right) \\ &+ \left\langle \log|\boldsymbol{K}_S + \sigma_S^2\boldsymbol{I}| \right\rangle_{p(\{\boldsymbol{x}_i\})} + P\log(2\pi) \Bigg). \end{aligned} \tag{72}$$

Clearly, $\left\langle \mathcal{L}_T(\mathcal{D}) \right\rangle_{p(\mathcal{D})}$ simplifies further:

$$\left\langle \mathcal{L}_T(\mathcal{D}) \right\rangle_{p(\mathcal{D})} = -\frac{1}{2}\left(P + \left\langle \log|\boldsymbol{K}_T + \sigma_T^2\boldsymbol{I}| \right\rangle_{p(\{\boldsymbol{x}_i\})} + P\log(2\pi) \right). \tag{73}$$

Then

$$\begin{aligned}\left\langle \mathcal{L}_T(\mathcal{D}) \right\rangle_{p(\mathcal{D})} - \left\langle \mathcal{L}_S(\mathcal{D}) \right\rangle_{p(\mathcal{D})} &= -\frac{1}{2}\Bigg(\mathrm{Tr}\left(\left\langle \left(\boldsymbol{K}_T + \sigma_T^2 \boldsymbol{I}\right)\left(\boldsymbol{K}_S + \sigma_S^2 \boldsymbol{I}\right)^{-1} \right\rangle_{p(\{\boldsymbol{x}_i\})} \right) \\ &\qquad + \left\langle \log \frac{|\boldsymbol{K}_T + \sigma_T^2 \boldsymbol{I}|}{|\boldsymbol{K}_S + \sigma_S^2 \boldsymbol{I}|} \right\rangle_{p(\{\boldsymbol{x}_i\})} + P \Bigg) \\ &= \Bigg\langle -\frac{1}{2}\Bigg(\mathrm{Tr}\left(\left(\boldsymbol{K}_T + \sigma_T^2 \boldsymbol{I}\right)\left(\boldsymbol{K}_S + \sigma_S^2 \boldsymbol{I}\right)^{-1} \right) \\ &\qquad + \log \frac{|\boldsymbol{K}_T + \sigma_T^2 \boldsymbol{I}|}{|\boldsymbol{K}_S + \sigma_S^2 \boldsymbol{I}|} + P \Bigg) \Bigg\rangle_{p(\{\boldsymbol{x}_i\})}\end{aligned} \tag{74}$$

which is just the averaged KL-divergence between the respective marginal likelihoods, as we wanted.

G Simulation details

All simulations were run on a MacBook Pro M1 2021 laptop, requiring no significant compute/memory resources. The most time-intensive simulations (Fig. 3) take 3-5 minutes to run.

The error curves in Fig. 2 were empirically generated using the closed form expressions for the posterior mean of the student, with data points sampled from the teacher prior. We considered dataset sizes ranging between $P \in [10, 100]$. We repeated simulations $C = 600$ times and averaged the results. The student and teacher noise models were assumed to be matching and were set to $\sigma_T = \sigma_S = 1.0$. The degrees of heterogeneity in the corresponding neural population comprised of neurons with Gaussian tuning curves exhibited varying bandwidths controlled by the α parameter displayed in the graph legends, where α parameterizes the inverse Gamma distribution from which widths are drawn, as described in the main text. In panels e-g, the homogeneous population is composed of neurons with Gaussian tuning curves with a shared bandwidth $\sigma = 1$. All heterogeneous populations that are used for comparison have an average bandwidth $\langle \sigma_n \rangle = 3$. Error bars in this experiment are reported as the SEM.

Panels b-e in Fig. 3 were generated using the `GaussianProcessRegressor` object provided by `sklearn`. The `.fit()` routine of the `GaussianProcessRegressor` object performed marginal likelihood optimization. Simulations were repeated for $C = 100$ runs. The teacher $\alpha = 0.2$. In some instances, marginal likelihood optimization failed, owing to the fact that the LML exhibits many local optima. Failure is indicated as a parameter reaching a preset bound (in the case of α, for example, bounds were set to 10^{-5} and 10^2). Such trials were recorded and discarded when computing held-out error. For all simulations, the student and teacher noise were assumed to be matching and set to $\sigma_T = \sigma_S = 0.02$. Error bars in this experiment are reported as SEM, owing to the fact that kernel optimization is over a landscape with many local optima, leading to highly variable results.

Figure 4 was generated without the use of the `GaussianProcessRegressor` object of `sklearn`. Instead, we simulated populations of Gaussian tuning curves, all of which shared a bandwidth $\sigma = 0.1$. The stimulus range was set to $x \in [-1, 1]$ and the reward peak was set to $r_{\max} = 0.5$. The student population's tuning centers were sampled according to $p(\mu) = \lambda \mathcal{N}(\mu_R, \sigma_R^2) + (1 - \lambda)\mathcal{U}(-L, L)$ where L is assumed to be larger than the stimulus range, such that that portion of the population yields a standard RBF kernel, $\mu_R = r_{\max}$ and $\sigma_R = 0.01$. The adapted tuning distribution was found by obtaining the estimate λ for the student population that maximized the marginal likelihood of a dataset sample. The early, middle and late designations in the figure correspond to training set size ($P = 10, 20/50, 100$, respectively). In the noise titrated case, the student's noise magnitude was scheduled to increase from $\sigma_S = 0.2$ (early) to 0.8 (late) as the dataset size increased. Otherwise, the noise model was matched to the teacher's. The teacher parameters where $\sigma_R^* = 0.01, \mu_R^* = r_{\max}, \sigma^* = 0.1, \sigma_T = 0.8$. Simulations were repeated for $C = 100$ runs. Error bars in this experiment are reported as SEM, owing to the fact that kernel optimization is over a landscape with many local optima, leading to highly variable results.

H Supplementary Figures

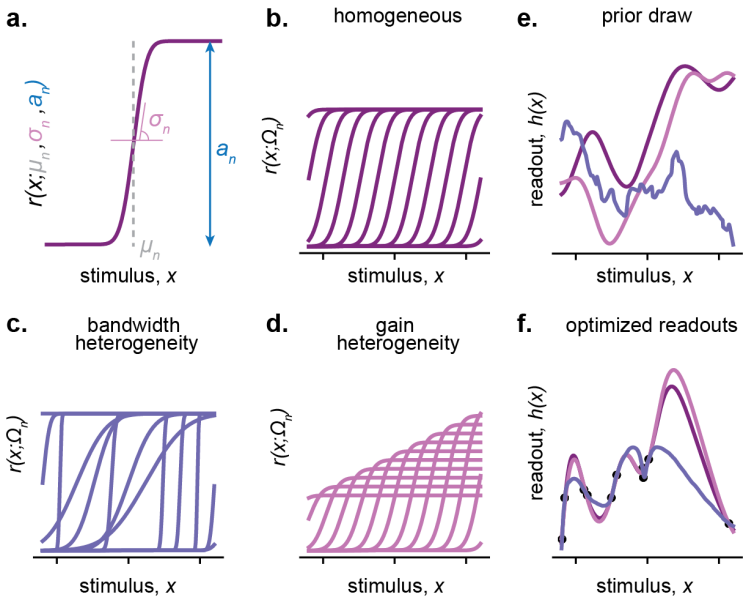

Figure S1: **Populations with sigmoidal tuning curves.** We demonstrate the inductive biases of populations with sigmoidal tuning curves, $r(x;\Omega_n) = a_n\,(\text{erf}((x-\mu_n)/\sigma_n)+1))$. The $+1$ bias term ensures that the neurons never have a negative firing rate. **a.** A schematic of how each parameter affects the shape of sigmoid tuning curves; a_n determines the peak firing rate, σ_n determines the steepness of the curve and μ_n determines the inflection point of the tuning curve. **b.** An example homogeneous population where $p(\mu_n) = \mathcal{U}(-L, L)$, $p(\sigma_n) = \delta(\sigma_n - \sigma)$ and $p(a_n) = \delta(a_n - g)$. **c.** An example population where $p(\sigma_n) = \text{Gamma}(\alpha, \sigma/\alpha)$ so that $\langle\sigma_n\rangle = \sigma$ and all other parameters are distributed as in **b.**. **d.** An example population where $p(a_n) \propto \mu_n$, ie., the gains increase as the tuning center increases in the stimulus space, and all other tuning parameters are the same as in **b.**. **e.** Example prior draws from each respective population, color coded. **f.** Example readouts from each population with optimally tuned weights for a given dataset (dots = observations).

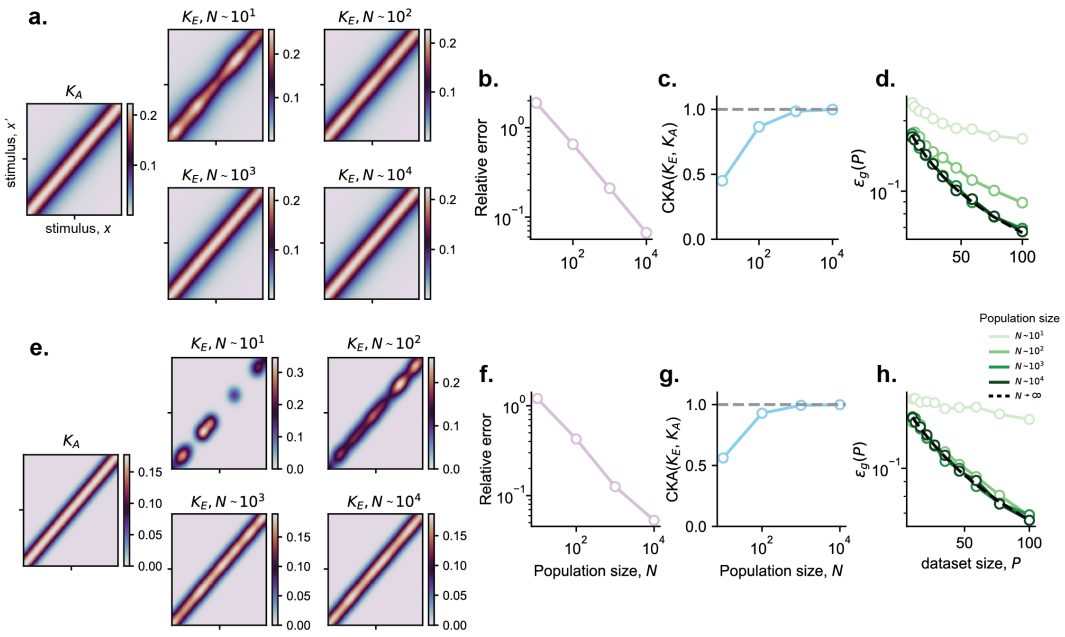

Figure S2: **Kernel structure and generalization error prediction accuracy converge for moderately sized populations. a.** Analytic rational quadratic (RQ) kernel ($\boldsymbol{K}_A$) and empirical kernel arising from a corresponding population with Gaussian tuning curves and Gamma-distributed tuning precisions ($\boldsymbol{K}_E$), with corresponding population size. **b.** The relative error ($\|\boldsymbol{K}_A - \boldsymbol{K}_E\|_F / \|\boldsymbol{K}_A\|_F$, lower=better) between the analytic and empirical kernels across population sizes. **c.** The centralized kernel alignment (CKA, [27–29]) metric between the empirical and analytic kernels (1 (gray doted line) = perfect kernel alignment, higher=better). **d.** Generalization error across dataset sizes for a student population with empirical kernels for various population sizes (kernels in **a**, population sizes various greens) and for the analytic kernel (black dotted line) for a teacher with an RQ kernel. **e-h.** The same as **a-d** but for an RBF kernel (and, correpsondingly, a population with Gaussian tuning curves and a shared homogeneous bandwidth).

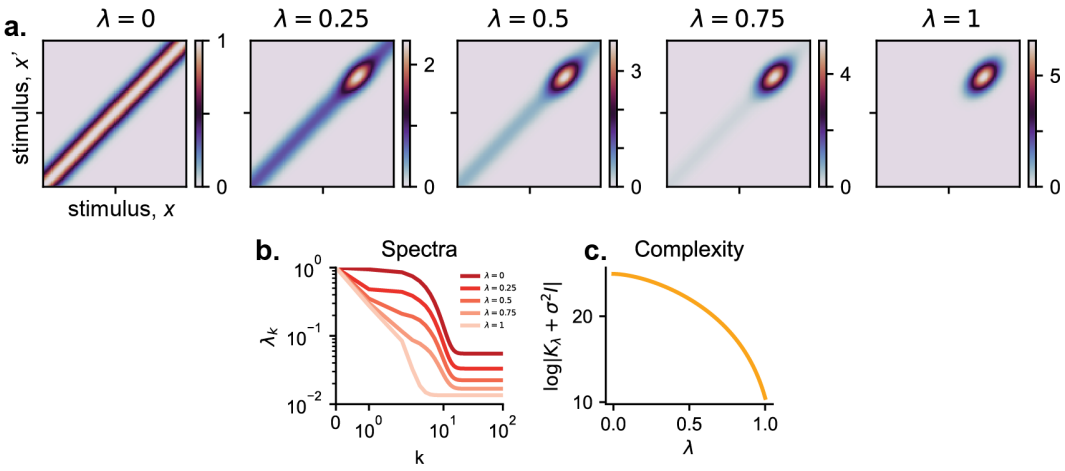

Figure S3: **Kernels induced by concentrated tuning centers are less complex than those induced by populations with uniform tuning centers.** In these figures, we show the kernels and associated complexity in populations in which $r(x;\Omega_n) = a_n \exp(-(\mu_n - x)^2/2\sigma_n^2)$ where $p(\sigma_n) = \delta(\sigma_n - 0.3)$, $p(\mu) = \lambda\mathcal{N}(1.5, 0.3) + (1-\lambda)\mathcal{U}(-L, L)$ where $L \gg x_{\max}$ and $p(a_n) = \delta(a_n - 1)$. **a.** Population kernels with varying degrees of tuning concentration, controlled by λ. **b.** Spectra of the respective kernels in the top row. As λ increases, the spectrum drops faster, indicating a simpler function space. **c.** The LML measure of kernel complexity, $\log|\boldsymbol{K}_\lambda + \sigma^2\boldsymbol{I}|$ (where $\sigma = 1$), as a function of λ, confirming a sharp drop-off in kernel complexity with greater tuning concentration, $\lambda \to 1$ as opposed to one with spatially homogeneous tuning, $\lambda \to 0$.

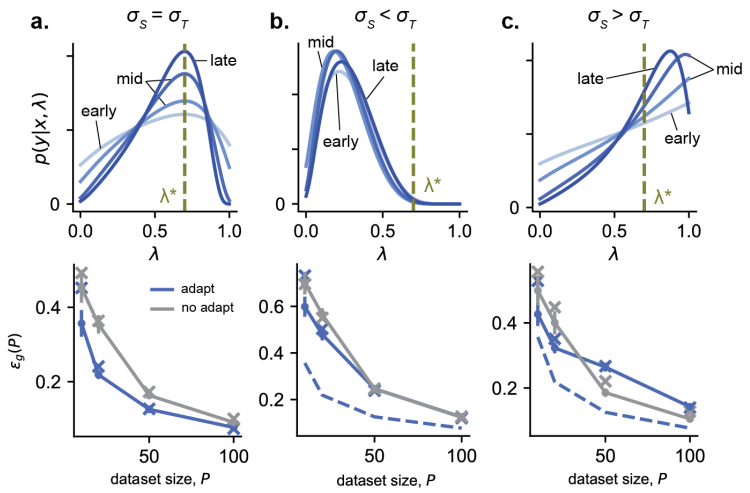

Figure S4: **The student noise model influences the degree of tuning concentration. a.** Result from the main text Fig. 4 for comparison. **b.** Adaptation when the student over-estimates the teacher noise model. Such a situation leads to an over-representation of the reward location by a significant degree that exceeds the optimal concentration as dictated by the teacher. Such populations incur significant additional error (bottom panel, dots = simulations, x = theory, blue dotted line = adapted population generalization error from panel **a**). **c.** Adaptation when the student under-estimates the teacher noise model. Such a situation leads to an under-representation of the reward location. Such populations incur significant additional error (bottom panel, dots = simulations, x = theory, blue dotted line = adapted population generalization error from panel **a**.).

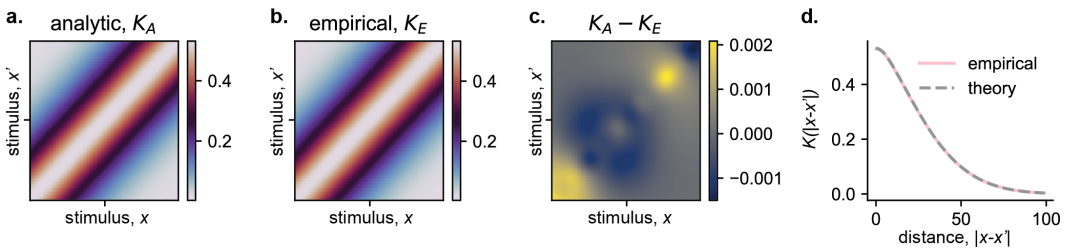

Figure S5: **The kernel for a population with a weighted mixture distribution of tuning induces a weighted mixture of population kernels**. Here, we consider a population where $p(\sigma_n) = \sum_{i=1}^{P} w_i \delta(\sigma_n - \sigma_i)$ where $w_i = 1/5 \; \forall i$ and $\{\sigma_i\} = [0.1, 0.3, 0.5, 0.7, 0.9]$, $p(a_n) = \delta(a_n - 1)$, $p(\mu) = \mathcal{U}(-L, L)$ where $L \gg x_{\max}$ and $r(x; \Omega_n) = a_n \exp(-|\mu_n - x|^2 / 2\sigma_n^2)$. Here we assume that the number of neurons in the population is $N = 1,000,000$. **a.** The analytic kernel computed as $\sum_{i=1}^{P} w_i K^{\text{RBF}}(\sigma_i)$, which is the analytic weighted mixture kernel. **b.** The empirical population kernel where $K_E(x, x') = \frac{1}{N} \boldsymbol{r}(x)^\top \boldsymbol{r}(x')$. **c.** The difference between the empirical and analytic kernels (note the color scale). **d**. A single slice of the kernel function (pink = empirical slice, grey = analytical slice).

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 claims that this paper explores the role of tuning heterogeneity in structuring priors for learning over distributions of functions. This claim is supported in Sec. 2. The abstract then claims that heterogeneous tuning that induces a prior well-calibrated to the distribution of target functions achieves the Bayes optimal error. This claim is supported in Sec. 3. The abstract then proposes that evidence maximization may be a normative objective whereby populations can minimize generalization error when the prior is poorly calibrated to the target distribution of functions. This claim is supported in Sec. 4. Finally, the abstract claims that our framework can explain the over-representation of rewarding areas by hippocampal neurons. This claim is supported in Sec. 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 primary limitations of the paper concern the biological plausibility of aspects of our model. For one, we assume deterministic tuning functions, while real neurons are stochastic in their firing. In addition, while we propose evidence maximization as a normative objective, we do not provide a biologically plausible implementation for doing so. These limitations are explained and meditated upon in the Discussion.

   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: [Yes]

   Justification: The main theoretical results are: (i) that tuning heterogeneity induces particular population kernels, (ii) a weighted mixture tuning distribution induces a population kernel that is a weighted mixture of the constituent tuning distributions, (iii) that the generalization error can be decomposed into an irreducible Bayes error and additional error arising from model mismatch, and (iv) that the difference in the average evidence between the student and teacher is equal to the average KL divergence between the respective marginal likelihoods. Derivations of all of these claims are provided in the Appendix.

   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: All information required to reproduce results in experiments (comprising Figs. 2,3,4) are provided in the Appendix, Sec. G.

   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: All experiments required to produce results discussed in this paper rely on existing, well-known libraries (such as `sklearn`, `numpy`, `scipy`). Our contributions do not constitute innovations on these libraries, nor do our results rely on any custom changes to these libraries. The code to reproduce all main text and supplementary figures can be found in https://doi.org/10.5281/zenodo.23984836.

   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: All hyper-parameters are specified for all experiments (see Sec. G).

   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: Error bars are provided in all simulation results. All errors are reported as ±SEM, indicated in figure captions.

   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: The time/memory requirements for all experiments are minimal, and can be run on a standard CPU of a MacBook Pro M1 2021 model. This is stated in Sec. G.

   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: This paper abides by the NeurIPS Code of Ethics. All guidelines of experimental reproduction are provided as faithfully as possible. We use no datasets that may compromise security or privacy. Finally, we disucss the broader impacts (see next item on checklist). Owing to the fact that this is a theoretical neuroscience paper, we expect no societal harms to emerge as a result of its publication.

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: [Yes]

    Justification: Broader impacts are discussed at the end of the Discussion. As this is a proposal of a neural theory, we expect no negative societal impacts as a result of this work. The positive societal impacts are an enhanced understanding of the role of tuning heterogeneity in biological neural systems.

    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: No datasets used in this paper pose safety risks. We do not anticipate that the model we discuss here is at any risk of being misused. As such, no safeguards are required.

    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: [N/A]

    Justification: All data used in experiments is synthetically generated.

    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: [N/A]

    Justification: We are not released new assets in this paper.

    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 crowdsource or human data is used in this paper.

    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 human participants were used in this study.

    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: LLMs were not used for any portion of this study.

    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.