Population Codes Enable Learning from Few Examples By Shaping Inductive Bias

Blake Bordelon
John A. Paulson School of
Engineering and Applied Sciences
Harvard University
Cambridge, MA

Cengiz Pehlevan
John A. Paulson School of
Engineering and Applied Sciences
Harvard University
Cambridge, MA

Abstract

The brain can learn from a limited number of experiences, an ability which requires suitable built in assumptions about the nature of the tasks which must be learned, or inductive biases. While inductive biases are central components of intelligence, how they are reflected in and shaped by population codes are not well-understood. To address this question, we consider biologically-plausible reading out of an arbitrary stimulus-response pattern from an arbitrary population code, and develop an analytical theory that predicts the generalization error of the readout as a function of the number of samples. We find that learning performance is controlled by the eigenspectrum of the population code's inner-product kernel, which measures the similarity of neural responses to two different input stimuli. Many different codes can realize the same kernel; by analyzing recordings from the mouse primary visual cortex, we demonstrate that biological codes are metabolically more efficient than other codes with identical kernels. We demonstrate that the spectral properties of the kernel introduce an inductive bias toward explaining stimulus-response samples with simple functions and determine compatibility of the population code with learning task, and hence the sample-efficiency of learning. While the tail of the spectrum is important for large sample size behavior of learning, for small sample sizes, the top eigenvalues of the spectrum govern generalization. We apply our theory to experimental recordings of mouse primary visual cortex neural responses, elucidating a bias towards sample-efficient learning of low frequency orientation discrimination tasks. We demonstrate this emergence of this bias in a simple model of primary visual cortex, and further show how invariances in the code to stimulus variations affect learning performance. Finally, we demonstrate that our methods are applicable to time-dependent neural codes. Overall, our study suggests sample-efficient learning as a general normative coding principle.

Introduction

The ability to learn fast is crucial for survival in a complex and an everchanging world, and the brain is remarkably efficient in this. Often, only a few experiences are sufficient to learn an ethologically relevant task, whether acquiring a new word [1] or recognizing a new face [2]. Despite the importance and ubiquity of sample efficiency with which our brains learn, our understanding of the brain's information encoding strategies that support this faculty remains poor [3, 4, 5].

In particular, when learning and generalizing from past experiences, and especially from few experiences, the brain relies on implicit assumptions it carries about the world, or its inductive biases [6, 7, 8, 5]. Reliance on inductive bias is not a choice: inferring a general rule from finite

observations is an ill-posed problem which requires prior assumptions since many hypotheses can explain the same observed experiences [9]. Consider learning a rule that maps photoreceptor responses to a prediction of whether an observed object is a threat or is neutral. Given a limited number of visual experiences of objects and their threat status, many threat-detection rules are consistent with these experiences. By choosing one of these threat-detection rules, the nervous system reveals an inductive bias. Without the right biases that suit the task at hand, successful generalization is impossible [6, 7, 8, 5]. Therefore, in order to understand why we learn certain tasks accurately and rapidly over others, we must understand the brain's inductive biases [3, 4, 5].

We address this question in a general neural circuit model which comprises of a population of sensory neurons and a readout neuron learning a stimulus-response pattern with a biologically-plausible learning rule (Fig 1A). In this circuit, inductive bias arises from the nature of the neural code for sensory stimuli. While different population codes can encode the same stimulus variables and allow learning of the same output with perfect performance given infinitely many samples, learning performance can depend dramatically on the code when restricted to a small number of samples, where the reliance on and the effect of inductive bias is strong (Fig 1B,C,D). Given the same stimulus-response examples, the readout neuron may make drastically different predictions depending on the nature of the code, leading to successful or failing generalizations (Fig 1C,D).

In order to understand how population codes shape inductive bias and allow fast learning of certain tasks over others, we develop an analytical theory of the readout neuron's learning performance as a function of the number of examples, or sample size. This theory illustrates how the structure of population codes constrain the learning system towards sample-efficient learning of certain functions over others, and how a match between the code and the task is crucial for sample-efficient learning. The readout's performance is completely determined by the code's kernel, a function which takes in pairs of population response vectors and outputs a representational similarity defined by the inner product of these vectors. While many codes could support the same kernel function, analyzing data from mouse primary visual cortex (V1) [10, 11, 12, 13], we find that biological code is metabolically more efficient than other possible codes that lead to the same kernel. Further, mouse V1 responses support sample-efficient learning of low frequency orientation discrimination tasks over high frequency ones. We demonstrate this bias in a simple model of V1 and show how response nonlinearity, sparsity, and relative proportion of simple and complex cells influence the code's bias and performance on learning tasks, including ones that involve invariances. Lastly, we show how these ideas can be extended to temporal population codes, using codes generated by recurrent neural networks in the context of a delayed match-to-sample task. Overall, our results suggest sample-efficient learning as a novel functional role for population codes.

Results

We consider a population of N neurons whose responses, $\{r_1(\boldsymbol{\theta}), r_2(\boldsymbol{\theta}), ..., r_N(\boldsymbol{\theta})\}$, vary with the input stimuli, which is in turn parameterized by a vector variable $\boldsymbol{\theta}$, such as the orientation and phase of a grating (Figure 1A). These responses define the population code. A readout neuron learns its set of weights $\mathbf{w}$ to approximate a stimulus-response map, or a target function $y(\boldsymbol{\theta})$, such as one that classifies stimuli as apetitive ($y = 1$) or aversive ($y = -1$), or a more smooth one that attaches intermediate values of valence. One important novelty of our theory is that it is general in its assumptions about the structure of the population code and the stimulus-response map considered (Methods), and can apply to many scenarios, including time varying codes.

The readout neuron learns from P stimulus-response examples with the goal of generalizing to

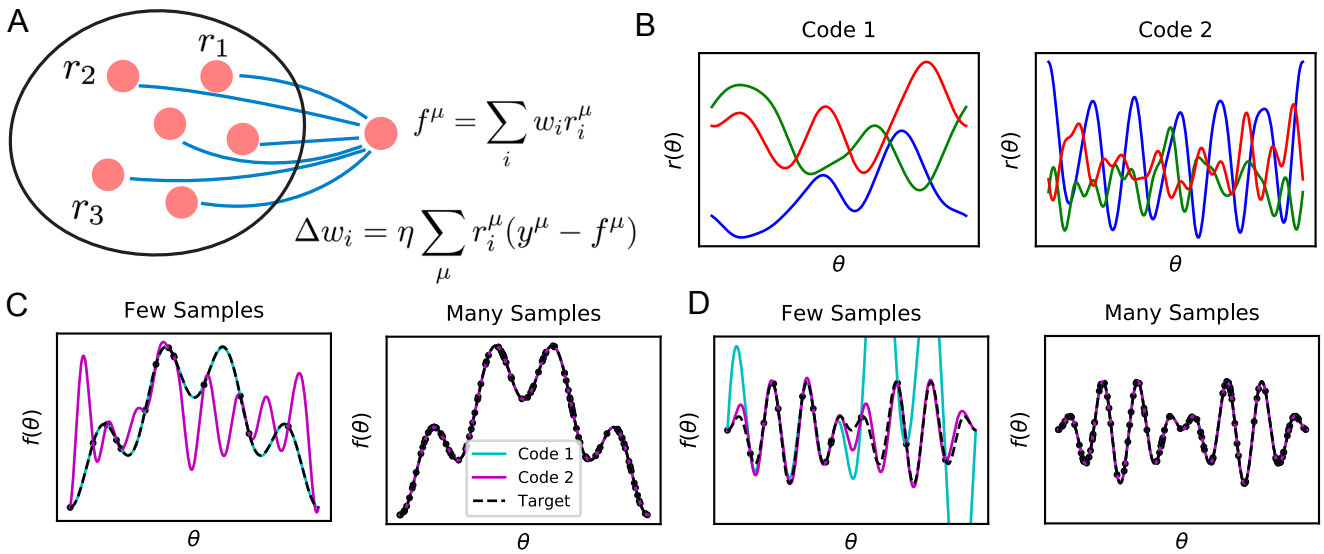

Figure 1: Learning tasks through linear readouts exploit representations of the population code to approximate a target response. **A** The readout weights from the population to a downstream neuron, shown in blue, are updated to fit target values y, using the local, biologically plausible delta rule. **B** Examples of tuning curves for two different population codes: Smooth tuning curves (Code 1) and rapidly varying tuning curves (Code 2). **C** (Left) A target function with low frequency content is approximated through the learning rule shown in **A** using these two codes. The readout from Code 1 (turquoise) fits the target function (black) almost perfectly with only $P = 12$ training examples, while readout from Code 2 (purple) does not accurately approximate the target function. (Right) However, when the number of training examples is sufficiently large ($P = 120$), the target function is estimated perfectly by both codes, indicating that both codes are equally expressive. **D** The same experiment is performed on a task with higher frequency content. (Left) Code 1 fails to perform well with $P = 12$ samples indicating mismatch between inductive bias and the task can prevent sample efficient learning while Code 2 accurately fits the target. (Right) Again, provided enough data $P = 120$, both models can accurately estimate the target function. Details of these simulations are given in Methods.

previously unseen examples. Example stimuli $\boldsymbol{\theta}^\mu$, $(\mu = 1, \ldots, P)$ are sampled from a probability distribution describing natural stimulus statistics $p(\boldsymbol{\theta})$. These stimuli and their target responses form a set of learning examples $\mathcal{D} = \{\boldsymbol{\theta}^\mu, y(\boldsymbol{\theta}^\mu)\}_{\mu=1}^P$. From this set, the readout weights are learned with the biologically-plausible delta-rule, $\Delta w_j = \eta \sum_\mu r_j(\boldsymbol{\theta}^\mu)(y(\boldsymbol{\theta}^\mu) - \boldsymbol{r}(\boldsymbol{\theta}^\mu) \cdot \mathbf{w})$, where η is a learning rate (Methods, Figure 1A). This learning process converges to a unique set of weights $\mathbf{w}^*(\mathcal{D})$ (Methods). Generalization error with these weights is given by

$$E_g(\mathcal{D}) = \int p(\boldsymbol{\theta})\ (\mathbf{w}^*(\mathcal{D}) \cdot \boldsymbol{r}(\boldsymbol{\theta}) - y(\boldsymbol{\theta}))^2 d\boldsymbol{\theta}, \tag{1}$$

which quantifies the expected error of the trained readout over the entire stimulus distribution $p(\boldsymbol{\theta})$. This quantity will depend on the population code $\boldsymbol{r}(\boldsymbol{\theta})$, the target function $y(\boldsymbol{\theta})$ and the set of training examples $\mathcal{D}$. Our theoretical analysis of this learning model provides insights into how populations of neurons encode information and allow sample-efficient learning.

Kernel structure of population codes controls learning

First, we note that the generalization performance of the learned readout on a given task depends entirely on the inner product kernel, defined by

$$K(\boldsymbol{\theta}, \boldsymbol{\theta}') = \frac{1}{N} \sum_{i=1}^{N} r_i(\boldsymbol{\theta}) r_i(\boldsymbol{\theta}'), \tag{2}$$

which quantifies the similarity of population responses to two different stimuli $\boldsymbol{\theta}$ and $\boldsymbol{\theta}'$, rather than the tuning curves of individual neurons. This follows from the fact that the learning procedure converges to the kernel regression solution $\mathbf{w}^*(\mathcal{D})$ for the training set $\mathcal{D}$ [14, 15], which implies that after identifying the solution, the readout neuron's learned output has the form

$$f(\boldsymbol{\theta}) = \mathbf{w}^*(\mathcal{D}) \cdot \boldsymbol{r}(\boldsymbol{\theta}) = \sum_{\mu=1}^{P} \alpha^\mu K(\boldsymbol{\theta}^\mu, \boldsymbol{\theta}), \tag{3}$$

where the coefficient vector $\boldsymbol{\alpha} = \mathbf{K}^+ \mathbf{y}$, where $^+$ denotes Moore-Penrose inverse (Methods), and the matrix $\mathbf{K}$ has entries $K_{\mu\nu} = K(\boldsymbol{\theta}^\mu, \boldsymbol{\theta}^\nu)$. The fact that the optimal solution can always be written as a linear combination of $\{K(\boldsymbol{\theta}, \boldsymbol{\theta}^\mu)\}_{\mu=1}^P$ is known as the representer theorem [16, 17]. Our main observation is that in these expressions the population code only appears through the kernel K. Therefore, the kernel controls the learned response pattern.

Biological codes are metabolically more efficient than other codes with identical kernels

The fact that learning performance depends only on the kernel introduces a large degeneracy in the set of codes which achieve identical desired performance on learning tasks. The kernel is invariant with respect to permutations of the neuron identity and left-rotations (of which permutations are a special case) of the population code. In particular, a population code $\boldsymbol{r}(\boldsymbol{\theta})$ can be rotated to generate a new code $\tilde{\boldsymbol{r}}(\boldsymbol{\theta})$ with identical kernel:

$$\tilde{\boldsymbol{r}}(\boldsymbol{\theta}) = \mathbf{Q}\boldsymbol{r}(\boldsymbol{\theta}), \tag{4}$$

where $\mathbf{Q}$ is an orthogonal matrix. Codes $\boldsymbol{r}(\boldsymbol{\theta})$ and $\tilde{\boldsymbol{r}}(\boldsymbol{\theta})$ will have identical linear readout performance on all possible learning tasks. We illustrate this degeneracy in Figure 2 using a publicly

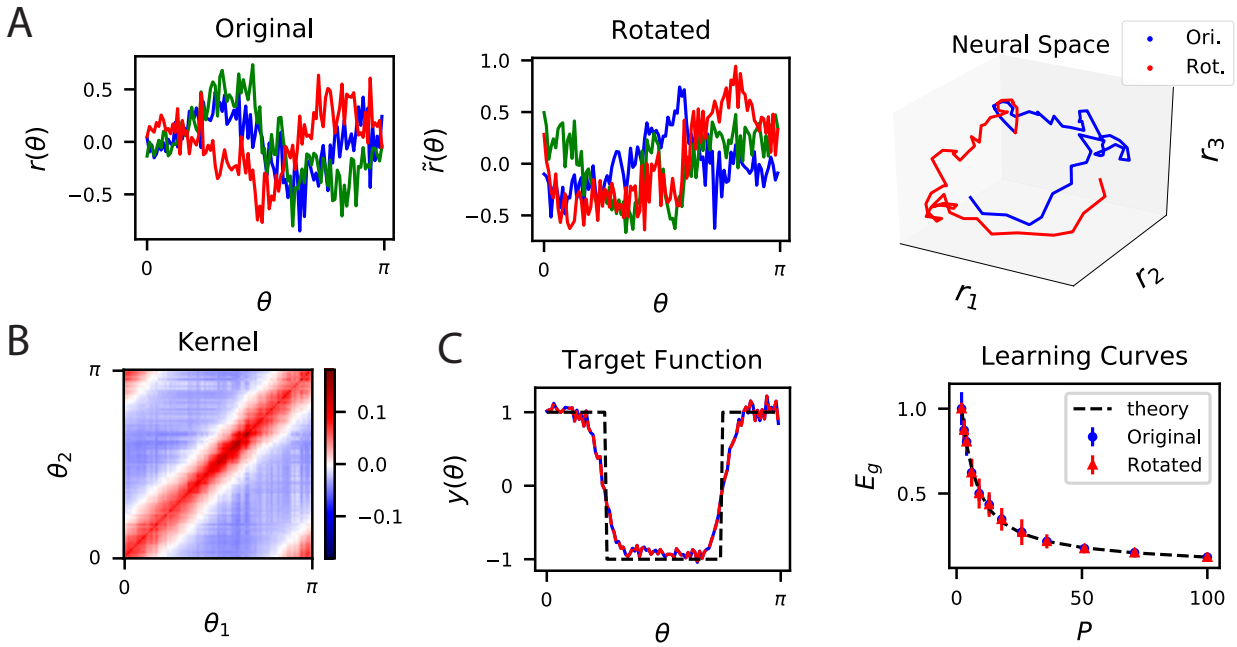

Figure 2: The inner product kernel controls the generalization performance of readouts. **A** Tuning curves $r(\theta)$ for three example recorded Mouse V1 neurons to varying static grating stimuli oriented at angle θ [10, 11] (Left) are compared with a randomly rotated version (Middle) $\tilde{\boldsymbol{r}}(\theta)$ of the same population code. (Right) These two codes, original (Ori.) and rotated (Rot.) can be visualized as parametric trajectories in neural space. **B** The inner product kernel matrix has elements $K(\theta_1, \theta_2)$. The original V1 code and its rotated counterpart have identical kernels. **C** In a learning task involving uniformly sampled angles, readouts from the two codes perform identically, resulting in identical approximations of the target function (shown on the left as blue and red curves) and consequently identical generalization performance as a function of training set size P (shown on right with blue and red points). The theory curve will be described in the main text.

available dataset which consists of activity recorded from $\sim$ 20,000 neurons from the primary visual cortex of a mouse while shown static gratings [10, 11]. An original code $\boldsymbol{r}(\boldsymbol{\theta})$ is rotated to generate $\tilde{\boldsymbol{r}}(\boldsymbol{\theta})$ (Figure 2A) which have the same kernels (Figure 2B) and the same performance on a learning task (Figure 2C).

Although, the performance of linear readouts may be invariant to such rotations, metabolic efficiency may favor certain codes over others [18, 19, 20, 21, 22], reducing degeneracy in the space of codes with identical kernels. We define $\boldsymbol{\delta}$ to be the vector of spontaneous firing rates of a population of neurons, and $\mathbf{s}^\mu = \boldsymbol{r}(\boldsymbol{\theta}^\mu) + \boldsymbol{\delta}$ be the spiking rate vector in response to a stimulus $\boldsymbol{\theta}^\mu$. The modulation with respect to the spontaneous activity, $\boldsymbol{r}(\boldsymbol{\theta}^\mu)$, gives the population code and defines the kernel, $K(\boldsymbol{\theta}^\mu, \boldsymbol{\theta}^\mu) = \frac{1}{N}\boldsymbol{r}(\boldsymbol{\theta}^\mu) \cdot \boldsymbol{r}(\boldsymbol{\theta}^\nu)$. To avoid confusion with $\boldsymbol{r}(\boldsymbol{\theta}^\mu)$, we will refer to $\mathbf{s}^\mu$ as total spiking activity. We propose that population codes prefer smaller spiking activity subject to a fixed kernel. In other words, because the kernel is invariant to any change of the spontaneous firing rates and left rotations of $\boldsymbol{r}(\boldsymbol{\theta})$, the orientation and shift of the population code $\boldsymbol{r}(\boldsymbol{\theta})$ should be chosen such that the resulting total spike count $\sum_{i=1}^{N}\sum_{\mu=1}^{P} s_i^\mu$ is small.

We tested whether biological codes exhibit lower total spiking activity than others exhibiting the same kernel on mouse V1 recordings, using deconvolved calcium activity as a proxy for spiking

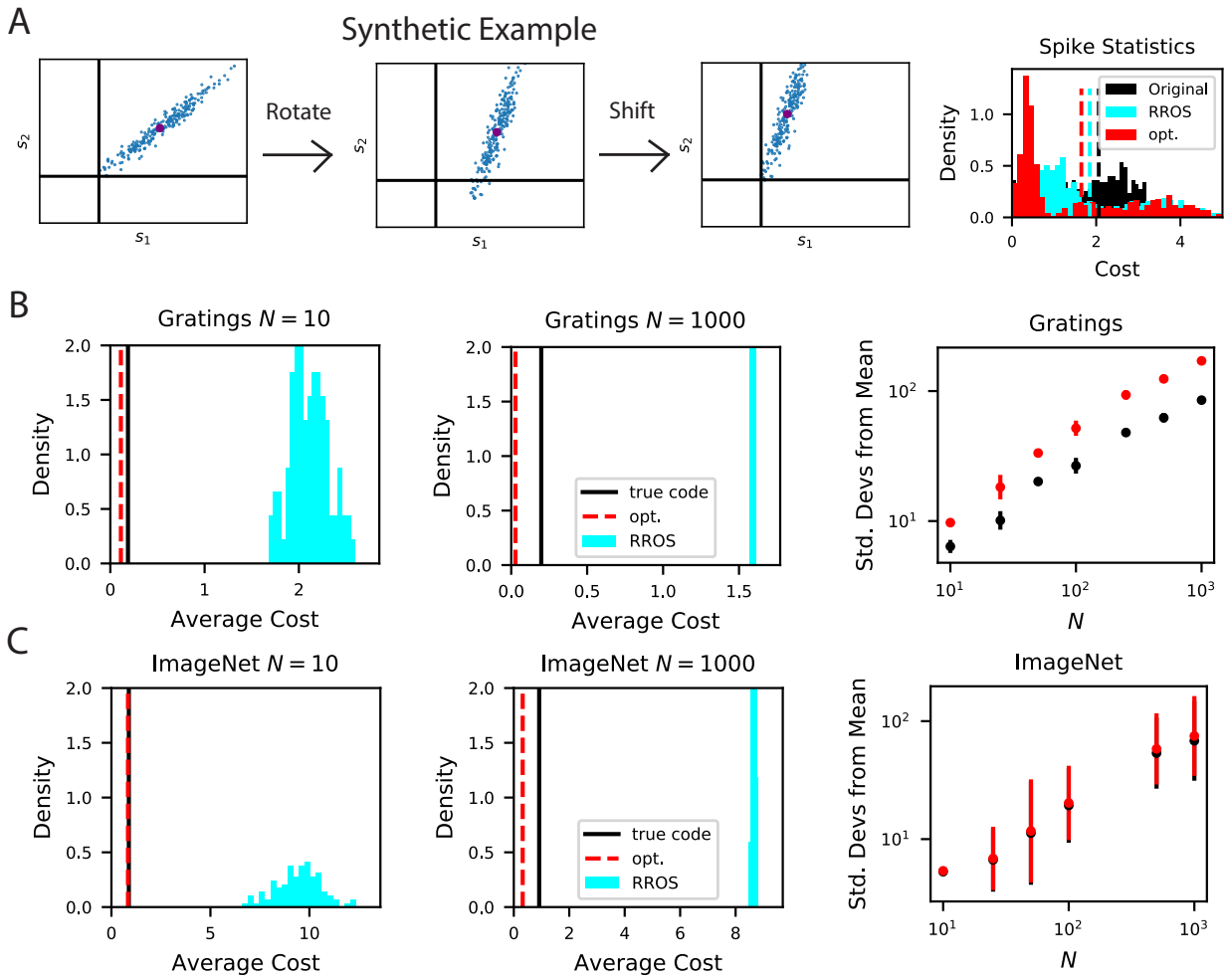

Figure 3: The biological code is more metabolically efficient than random codes with same inductive biases. **A** We illustrate our procedure in a synthetic example. A non-negative population code (left) can be rotated about its spontaneous firing rate (middle), illustrated as a purple dot, and shifted to a new non-negative population code (right). If the kernel is measured about the spontaneous firing rate, these transformations leave the inductive bias of the code invariant but can change the total spiking activity of the neural responses. We refer to such an operation as random rotation + optimal shift (RROS). We also perform gradient descent over rotations and shifts, generating an optimized code (opt). **B** Performing RROS on N neuron subsamples of experimental Mouse V1 recordings [10, 11], shows that the true code has much lower average cost $\frac{1}{NP}\sum_{i\mu} s_i^\mu$ compared to random rotations of the code. The set of possible RROS transformations (Methods) generates a distribution over average cost, which has higher mean than the true code. We also optimize metabolic cost over the space of RROS transformations, which resulted in the red dashed lines. We plot the distance (in units of standard deviations) between the cost of the true and optimal codes and the cost of randomly rotated codes for different neuron subsample sizes N. **C** The same experiment performed on Mouse V1 responses to ImageNet images from 10 relevant classes [13, 12].

events [10, 11, 23] (Figure 3). To compare the experimental total spiking activity to other codes with identical kernels, we computed random rotations of the neural responses around spontaneous activity, $\tilde{\boldsymbol{r}}(\boldsymbol{\theta}^\mu) = \mathbf{Q}\boldsymbol{r}(\boldsymbol{\theta}^\mu)$, and added the $\tilde{\boldsymbol{\delta}}$ that minimizes total spiking activity and maintains its nonnegativity (Methods). In other words, we compare the true code to the most metabolically efficient realizations of its random rotations. This procedure may result in an increased or decreased total spike count in the code, and is illustrated in a synthetic dataset in Figure 3A. We conducted this procedure on subsets of various sizes of mouse V1 neuron population code, as our proposal should hold for any subset of neurons (Methods), and found that the true V1 code is much more metabolically efficient than randomly rotated versions of the code (Figure 3 B and C). The large disparity in total spiking activity between the true and randomly generated codes with identical kernels suggests that metabolic constraints may break the rotational symmetry in coding. This finding holds for both responses to static gratings and to natural images as we show in Figure 3 B and C respectively.

To further explore the idea of metabolic efficiency as a normative principle, we posed an optimization problem which identifies the most metabolically efficient code with the same kernel as the biological V1 code. This optimization problem searches over rotation matrices $\mathbf{Q}$ and finds the $\mathbf{Q}$ matrix and off-set vector $\boldsymbol{\delta}$ which gives the lowest cost $\sum_{i\mu} s_i^\mu$ (Methods)(Figure 3). Though the local optimum identified with the algorithm is lower in cost than the biological code, both the optimal and biological codes are significantly displaced from the distribution of random codes with same kernel. Our findings do not change when data is preprocessed with an alternative strategy, an upper bound on neural responses is imposed on rotated codes, or subsets of stimuli are considered (Supplementary Information (SI) and Figure SI.1). Overall, these observations suggest that metabolic efficiency may favor certain codes over others that realize the same kernel.

Code-task alignment governs generalization

We next examine how the population code affects generalization performance of the readout. We calculated analytical expressions of the average generalization error in a task defined by the target response $y(\boldsymbol{\theta})$ after observing P stimuli using methods from statistical physics (Methods). As discussed above, the relevant quantity in learning performance is the kernel, allowing us to leverage results from our previous work studying generalization in kernel regression [24, 25]. Specifically, we calculated the generalization error averaged over all possible realizations of the training dataset of composed of P stimuli drawn from the stimulus distribution, $\langle E_g \rangle_{\mathcal{D}}$ (Equation (25)). As P increases, the variance in E_g due to the composition of the dataset falls, and our expressions become descriptive of also the typical case. We provide details of our calculations in Methods and SI.

One of our main observations is that given a population code $\boldsymbol{r}(\boldsymbol{\theta})$, the singular value decomposition of the code gives the appropriate basis to analyze the inductive biases of the readouts (Figure 4A). The tuning curves for individual neurons $r_i(\boldsymbol{\theta})$ for stimulus variable $\boldsymbol{\theta}$ form an N-by-M matrix $\boldsymbol{R}$, where M, possibly infinite, is the number of all possible stimuli. The left-singular vectors (or principal axes) and singular values of this matrix have been used in neuroscience for describing lower dimensional structure in the neural activity and estimating its dimensionality, see e.g. [26, 27, 28, 29, 30, 31, 13, 10, 32, 33, 34]. We found that the function approximation properties of the code are controlled by the singular values, or rather their squares $\{\lambda_k\}$ which give variances along principal axes, indexed in decreasing order, and the corresponding right singular vectors $\{\phi_k(\boldsymbol{\theta})\}$, also known as kernel eigenfunctions (Methods and SI). This follows from the fact that learned response (Eq. (3)) is only a function of the kernel K, and the eigenvalues λ_k and orthonormal eigenfunctions $\phi_k(\boldsymbol{\theta})$ collectively define the code's inner-product kernel $K(\boldsymbol{\theta}, \boldsymbol{\theta}')$ through

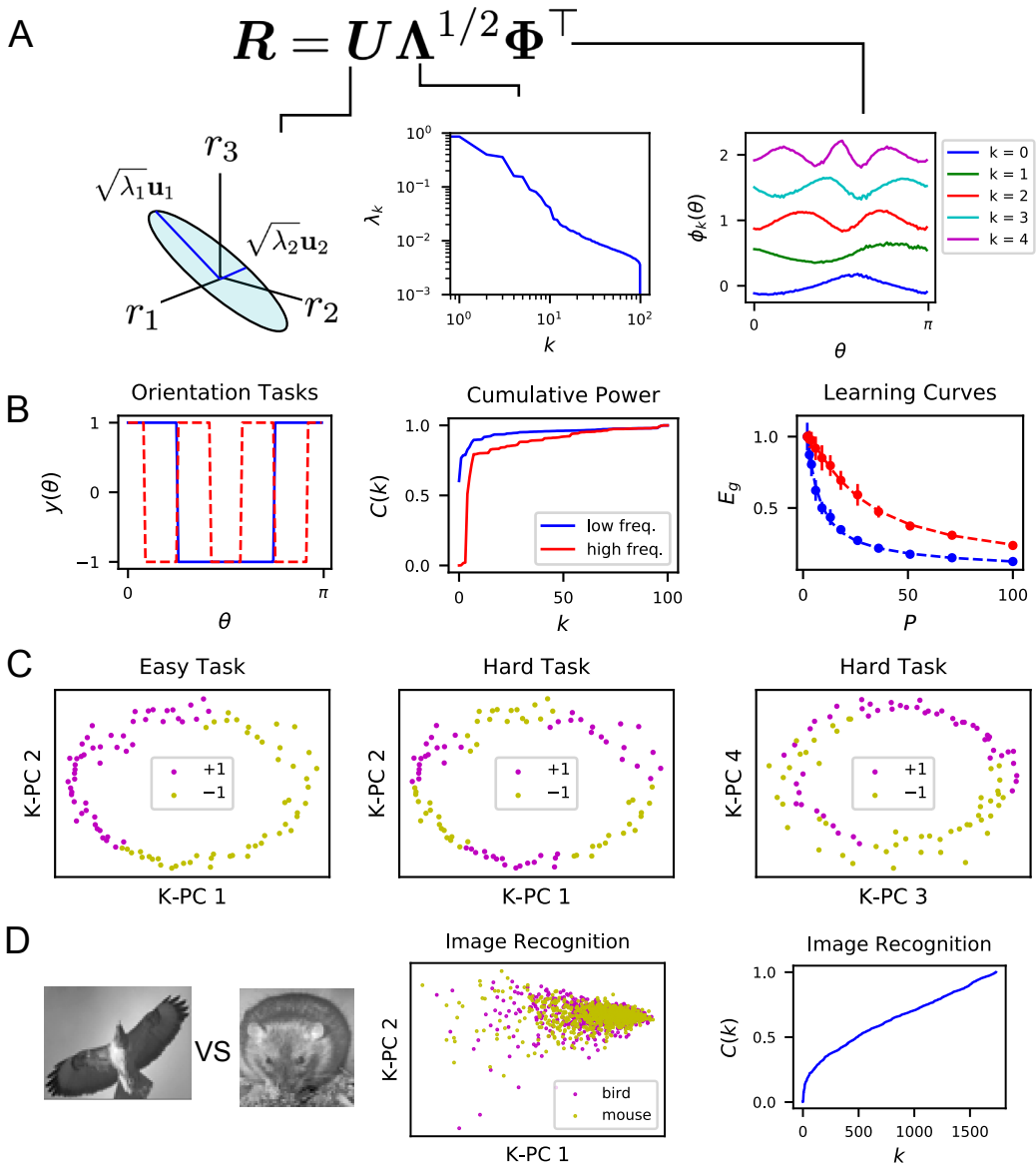

Figure 4: The singular value decomposition of the population code reveals the structure and inductive bias of the code. **A** Singular value decomposition of the response matrix $\mathbf{R}$ gives left singular vectors $\mathbf{u}_k$ (principal axes), kernel eigenvalues λ_k, and kernel eigenfunctions $\phi_k(\boldsymbol{\theta})$. The ordering of eigenvalues provides an ordering of which modes ϕ_k can be learned by the code from few training examples. The eigenfunctions were offset for visibility. **B** (Left) Two different learning tasks $y(\theta)$, a low frequency (blue) and high frequency (red) function, are shown. (Middle) The cumulative power distribution rises more rapidly for the low frequency task than the high frequency, indicating better alignment with top kernel eigenfunctions and consequently more sample-efficient learning as shown in the learning curves (right). Dashed lines show theoretical generalization while dots and solid vertical lines are experimental average and standard deviation over 30 repeats. **C** The feature space representations of the easy (left) and hard (middle and right) tasks. Each point represents the embedding of a response along the k-th principal axis $\boldsymbol{r}^\mu \cdot \mathbf{u}_k = \sqrt{\lambda_k}\phi_k(\boldsymbol{\theta}^\mu)$. The binary target value $\{\pm 1\}$ is indicated with the color of the point. The easy (left), low frequency task is well separated along the top two dimensions, while the hard task is not linearly separable in two (middle) or even with four feature dimensions (right). **D** On an image discrimination task (recognizing birds vs mice, left), V1 has an entangled representation which does not allow good performance of linear readouts. This is evidenced by the projection of the responses along the top K-PCs (middle) and the slowly rising $C(k)$ curve (right).

an eigendecomposition $K(\boldsymbol{\theta}, \boldsymbol{\theta}') = \frac{1}{N}\sum_{i=1}^{N} r_i(\boldsymbol{\theta}) r_i(\boldsymbol{\theta}') = \sum_k \lambda_k \phi_k(\boldsymbol{\theta}) \phi_k(\boldsymbol{\theta}')$ [35] (Methods and SI).

Our analysis showed the existence of a bias in the readout towards learning certain target responses faster than others. The kernel eigenfunctions form a complete basis for square integrable functions, allowing the expansion of the target response $y(\boldsymbol{\theta}) = \sum_k v_k \phi_k(\boldsymbol{\theta})$ and the learned readout response $f(\boldsymbol{\theta}) = \sum_k \hat{v}_k(\mathcal{D}) \phi_k(\boldsymbol{\theta})$ in this basis. We found that the readout's generalization is better if the target function $y(\boldsymbol{\theta})$ is aligned with the top eigenfunctions ϕ_k, equivalent to v_k^2 decaying rapidly with k (Methods). We formalize this notion by the following metric. Mathematically, generalization error $\langle E_g \rangle$ can be decomposed into normalized estimation errors E_k for the coefficients of these eigenfunctions ϕ_k, $\langle E_g \rangle_{\mathcal{D}} = \sum_k v_k^2 E_k$, where $E_k = \left\langle (\hat{v}_k(\mathcal{D}) - v_k)^2 \right\rangle_{\mathcal{D}} / v_k^2$. We find that the ordering of the eigenvalues λ_k controls the rates at which these mode errors E_k decrease as P increases (Methods):

$$\lambda_k > \lambda_\ell \implies E_k < E_\ell. \tag{5}$$

Hence, larger eigenvalues mean lower generalization error for those normalized mode errors E_k, indicating a *spectral bias* of the readout.

Based on this observation, we propose *code-task alignment* as a principle for good generalization. To quantify code-task alignment, we use a cumulative power distribution $C(k)$ which measures the total power in of the target function in the top k eigenmodes, normalized by the total power [25]:

$$C(k) = \frac{\sum_{\ell=1}^{k} v_\ell^2}{\sum_{\ell=1}^{\infty} v_\ell^2}. \tag{6}$$

Functions that have high alignment with the population code's kernel will have quickly rising cumulative power distributions $C(k)$, since a large proportion of power is placed in the top modes. Within our theory, target functions with high $C(k)$ can be learned with fewer training samples than target functions with low $C(k)$ since the mode errors E_k are ordered for all P (Methods).

This theory can be used to probe the learning biases of neural populations. Using publicly available calcium imaging recordings from mouse primary visual cortex (V1), we analyzed population responses to static grating stimuli oriented at an angle θ [10, 11]. We found that the kernel eigenfunctions have sinusoidal shape with differing frequency. The ordering of the eigenvalues and eigenfunctions in Figure 4A indicates a frequency bias: lower frequency functions of θ are easier to estimate at small sample sizes.

We tested this idea by constructing two different orientation discrimination tasks shown in Figures 4B,C. We trained the readout using a subset of the experimentally measured neural responses, and measured the readout's generalization performance. Intuitively, we would expect that the low frequency orientation task to be easier for the mouse to learn at low sample sizes than the high frequency orientation target function. We found that the cumulative power distribution for the low frequency task has a more rapidly rising $C(k)$ (Figure 4B). Using our theory of generalization, we can predict learning curves for these two tasks, which express the generalization error as a function of the number of sampled stimuli P. The error for the easy task is lower at all sample sizes than the hard task. The theoretical predictions and numerical experiments show perfect agreement (Figure 4B). More intuition can be gained by visualizing tasks in feature space. For the easy task, the two target values are well separated along the top two kernel eigenfunctions. However, the hard task is not well separated along even the top four features (Figure 4C).

Using the same ideas, we can use our theory to get insight into tasks which the V1 population code is ill-suited to learn through linear readout. For the task of identifying mice and birds [13, 12] the linear rise in cumulative power indicates that there is roughly equal power along all kernel eigenfunctions, indicative of a representation poorly aligned to this task. (Figure 4D)

Low frequency bias and code-task alignment in a simple model of V1

Next, we study a simple model of V1 to elucidate factors that lead to the low frequency bias. We model responses of V1 neurons as photoreceptor inputs passed through Gabor filters and a subsequent nonlinearity, $\psi(z)$, modeling a population of orientation selective simple cells (Figure 5 A) (Methods and SI). In this model, the kernel for static gratings with orientation $\theta \in [0, \pi]$ is circulant $K(\theta, \theta') = \kappa(\theta - \theta')$, and, as a consequence, the eigenfunctions of the kernel in this setting are Fourier modes (Methods). The eigenvalues, and hence the strength of the spectral bias, are determined by the nonlinearity.

Motivated by findings in the primary visual cortex [36, 37, 38, 39], we studied the spectral bias induced by rectified power-law nonlinearities of the form $\psi(z) = \max\{0, z - a\}^q$, where the parameter a controls the sparsity of the code and the parameter q controls the smoothness of the tuning curve about $z = a$. We fit q and a to the Mouse V1 kernel (black dots) and show the result of the fit (blue) and comparisons to other parameter sets in Figure 5B. Computation of the kernel and its eigenvalues (Methods) indicates a low frequency bias: the eigenvalues for low frequency modes are higher than those for high frequency modes, indicating a strong inductive bias to learn functions of low frequency in the orientation. Decreasing sparsity (a) leads to an faster decay in the spectrum and a stronger bias towards lower frequency functions (Figure 5B; more comparisons in Figure SI.2). The effect of the power of nonlinearity q is more nuanced: increasing power may increase spectra at lower frequencies, but may also lead to a faster decay at the tail (Figure 5B; more comparisons in Figure SI.2). The behavior at low frequencies may have significant impact for learning with few samples. We discuss this in more detail in the next section. Overall, our findings show that the spectral bias of a population code can be determined in non-trivial ways by its biophysical parameters, including neural thresholds and nonlinearities.

To further illustrate the importance of code-task alignment, we next study how invariances in the code to stimulus variations may affect the learning performance. We introduce complex cells in addition to simple cells in our model with proportion $s \in [0, 1]$ of simple cells (Methods; Figure 5A), and allow phase, ϕ, variations in static gratings. We use the energy model [40, 41] to capture the phase invariant complex cell responses (Methods). We reason that in tasks that do not depend on phase information, complex cells should improve sample efficiency. In this model, the kernel for the V1 population is a convex combination of the kernels for the simple and complex cell populations

$$K_{V1}(\theta, \theta', \phi, \phi') = sK_s(\theta, \theta', \phi, \phi') + (1 - s)K_c(\theta, \theta'), \tag{7}$$

where K_s is the kernel for a pure simple cell population that depends n both orientation and phase, and K_c is the kernel of a pure complex cell population that is invariant to phase (Methods, Eqs. (34) and (49)). Figure 5C shows top kernel eigenfunctions for various values of s elucidating inductive bias of the readout. Figures 5D and 5E show that on pure orientation discrimination tasks, increasing the proportion of complex cells by decreasing s improves generalization. This is intuitively sensible since we expect increasing the sensitivity to the nuisance variable ϕ would only degrade performance on a task only involving θ. The cumulative power distribution is also maximized at $s = 0$. However, on a task which only depends on the phase variable ϕ, a pure complex cell population cannot generalize, since variation in the target function due to changes in ϕ cannot be explained in the codes' responses. In this setting, a pure simple cell population is required to attain optimal performance, $s = 1$. The cumulative power distribution is maximized at $s = 1$. Lastly, we consider a nontrivial hybrid task which requires utilization of both variables θ, ϕ. In this setting, an optimal mixture s exists for each sample budget P which minimizes the generalization error. The cumulative power distribution is maximized at different s values depending on k, the component of the target function. This is consistent with an optimal heterogenous mix, because

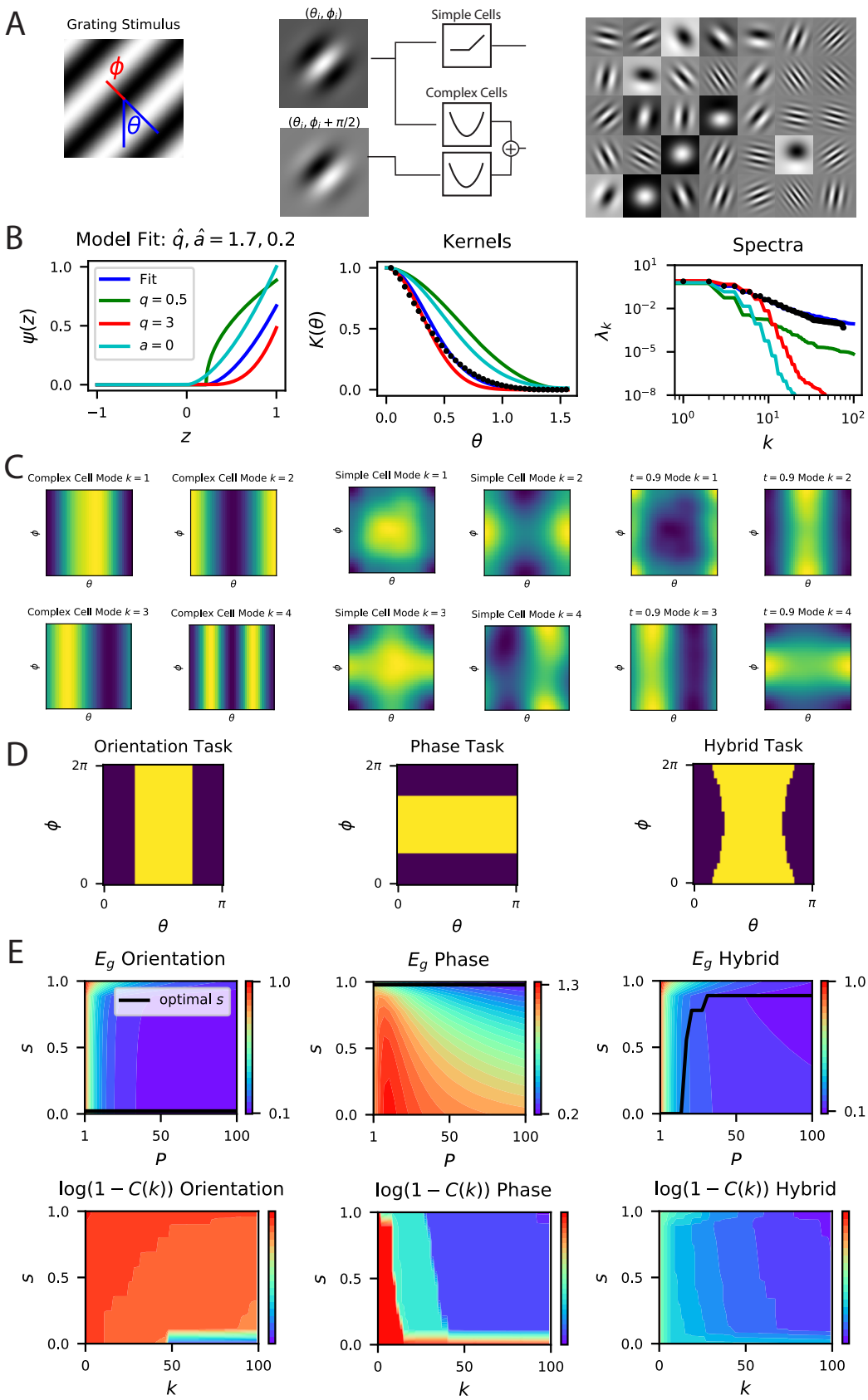

Figure 5: Caption on next page

Figure 5: A model of V1 as a bank of Gabor filters recapitulates experimental inductive bias. **A** Gabor filtered inputs are mapped through nonlinearity. A grating stimulus (left) with orientation θ and phase ϕ is mapped through a circuit of simple and complex cells (middle). **B** A threshold-powerlaw nonlinearity $\psi_{q,a}(z) = \max\{0, z-a\}^q$ is fit to the mouse V1 kernel (black dots). Kernels and spectra for alternative choices of q, a are shown. **C** We model V1 as a mixture of sN simple cells and $(1-s)N$ complex cells. A pure complex cell population has all eigenfunctions independent of phase ϕ. A pure simple cell population $s = 1$ or mixture codes $0 < s < 1$ depend on both orientation phase in a nontrivial way. **D** Three tasks are visualized, where yellow represents $y(\theta, \phi) = +1$ and purple represents $y(\theta, \phi) = -1$. The left task only depends on orientation stimulus variable θ, the middle only depends on phase ϕ, the hybrid task (right) depends on both. **E** (top) Generalization error and cumulative power distributions for the three tasks as a function of the simple-complex cell mixture parameter s. In Figure SI.2 we provide more comparisons of our theory and numerical experiments.

components of the target are learned successively with increasing sample size. We conclude that the degree of invariance required for the set of natural tasks, and the number of samples determine the optimal simple cell, complex cell mix.

Small and large sample size behaviors of generalization

Our results so far point to the fact that generalization with low sample sizes crucially depend on the top eigenvalues and eigenfunctions of the code's kernel. This is to be contrasted with a recent proposal about the effect of asymptotic decay rate of the kernel eigenvalues on generalization. Stringer *et al.* [13] argued that the input-output differentiability of the code may be necessary for better generalization, which is in turn governed by the asymptotic rate of spectral decay. Here, we provide an example to illustrate that asymptotic conditions on the kernel spectrum are insufficient to provide generalization guarantees when the sample size is small.

Our first example demonstrates how a code allowing good generalization for large sample sizes can be disadvantageous for small sizes. In Figure 6A, we plot three different populations of neurons with smooth (infinitely differentiable) tuning curves that tile a periodic stimulus variable, such as the direction of a moving grating. The tuning width, σ, of the tuning curves strongly influences the structure of these codes: narrower bandwidths have more high frequency content as we illustrate in a random 3D projection of the population code for $\theta \in [0, 2\pi]$ (Figure 6A). Visualization of the corresponding (von Mises) kernels and their spectra are provided in Figure 6B. The width of the tuning curves control bandwidths of the kernel spectra Figure 6B, with narrower curves having an later decay in the spectrum and higher high frequency eigenvalues. These codes can have dramatically different generalization performance, which we illustrate with a simple "bump" target function (Figure 6C). In this example, for a sample size of $P = 10$, we observe that codes with too wide or too narrow tuning curves (and kernels) do not perform well, and there is a well-performing code with an optimal tuning curve width σ, which is compatible with the width of the target bump, σ_T. We find that in general the optimal σ is different for each P (Figure 6C). In the large P regime, the ordering of the performance of the three codes are reversed (Figure 6D). In this regime, generalization error scales in a power law $E_g \sim P^{-\min\{2, \sigma_T/\sigma\}}$ and the narrow code, which performed worst for $P \sim 10$, performs the best. This example demonstrates that asymptotic conditions on the tail of the spectra are insufficient to understand generalization in the small sample size limit. The bulk of the kernel's spectrum needs to match the spectral structure of the task at hand to generalize efficiently in the low-sample size regime. However, for large sample sizes, the

tail of the eigenvalue spectrum becomes important.

We repeat the same exercise for a Laplace kernel showing that this is not an artifact of the infinite differentiability of von Mises kernels. Each of these Laplace kernels has the same asymptotic power law spectrum $\lambda_k \sim o(k^{-2})$, as observed in recordings of [13], exhibiting a discontinuous first derivative (Figure 6E). Despite having the same spectral scaling at large k, these kernels can give dramatically different performance in learning tasks, again indicating the influence of the top eigenvalues on generalization at small P (Figure 6F). Again, the trend for which kernels perform best at low P can be reversed at large P. In this case, all generalization errors scale with $E_g \sim P^{-2}$ (Figure 6G). More generally, our theory shows that if the task power spectrum and kernel eigenspectrum are both falling as power laws with exponents a and b respectively, then the generalization error asymptotically falls with a power law, $E_g \sim P^{-\min(a-1,2b)}$ [24] (Methods). This decay is fastest when $b \geq \frac{a-1}{2}$ for which $E_g \sim P^{-(a-1)}$. Therefore, the tail of the kernel's eigenvalue spectrum determines the large sample size behavior of the generalization error for power law kernels.

Time-Dependent Neural Codes

Our discussion so far focused on static stimuli and readouts, however our framework can directly be extended to learning of arbitrary time-varying functions of time-varying inputs from an arbitrary temporal population code (Methods). In this setting, the population code $\boldsymbol{r}(\{\boldsymbol{\theta}(t)\}, t)$ is a function of an input stimulus sequence $\boldsymbol{\theta}(t)$ and possibly its entire history, and time t. A downstream linear readout $f(\{\boldsymbol{\theta}\}, t) = \mathbf{w} \cdot \boldsymbol{r}(\{\boldsymbol{\theta}\}, t)$ learns a target sequence $y(\{\boldsymbol{\theta}\}, t)$ from a total of $\mathcal{P}$ examples that can come at any time during any sequence. Learning is again achieved through the delta-rule and the learned function can be expressed as a linear combination of the kernel evaluated at the $\mathcal{P}$ examples. The kernel in this case is a more complicated object that computes inner products of neural population vectors at different times t, t' for different input sequences $\{\boldsymbol{\theta}\}, \{\boldsymbol{\theta}'\}$: $K(\{\boldsymbol{\theta}\}, \{\boldsymbol{\theta}'\}, t, t') = \frac{1}{N}\boldsymbol{r}(\{\boldsymbol{\theta}\}, t) \cdot \boldsymbol{r}(\{\boldsymbol{\theta}\}, t')$ [42, 43, 44]. Our theory carries over from the static case with appropriate modifications (Methods). Kernels whose top eigenfunctions have high alignment with the target dynamical system $y(\{\boldsymbol{\theta}\}, t)$ will achieve the best average case generalization performance.

As a concrete example, we focus on readout from a temporal population code generated by recurrent neural networks in a delayed match-to-sample task (Figure 7A) [45, 46]. We study a rate based model of randomly connected recurrent network dynamics which obeys $\tau\dot{\mathbf{u}}(t) = -\mathbf{u}(t) + \frac{g}{\sqrt{N}}\mathbf{W}_{\boldsymbol{r}}\boldsymbol{r}(t) + \mathbf{W}_{\boldsymbol{\theta}}\boldsymbol{\theta}(t)$ where the rates are related to input currents through a tanh nonlinearity $\boldsymbol{r}(t) = \tanh(\mathbf{u}(t))$. The gain variable g was set to 1.5 to generate a rich dynamics in the network [47]. In this task, the network is presented for a short time an input cue sequence coding an angular variable which is drawn randomly from a distribution (Figure 7B). The recurrent neural network must remember this angle and reproduce an output sequence which is a simple step function whose height depends on the angle which begins after a time delay from the cessation of input stimulus and lasts for a short time (Figure 7C). Kernel induced by the spatiotemporal code and its eigensystem are shown in Figure 7D and E. The high dimensional nature of the activity in the recurrent network introduces complex and rich spatiotemporal similarity structure. With enough samples, the readout neuron can learn to output the desired angle with high fidelity Figure 7F. Unsurprisingly, tasks involving long time delays are more difficult and exhibit lower cumulative power curves. Consequently, the generalization error for small delay tasks drops much more quickly with increasing $\mathcal{P}$.

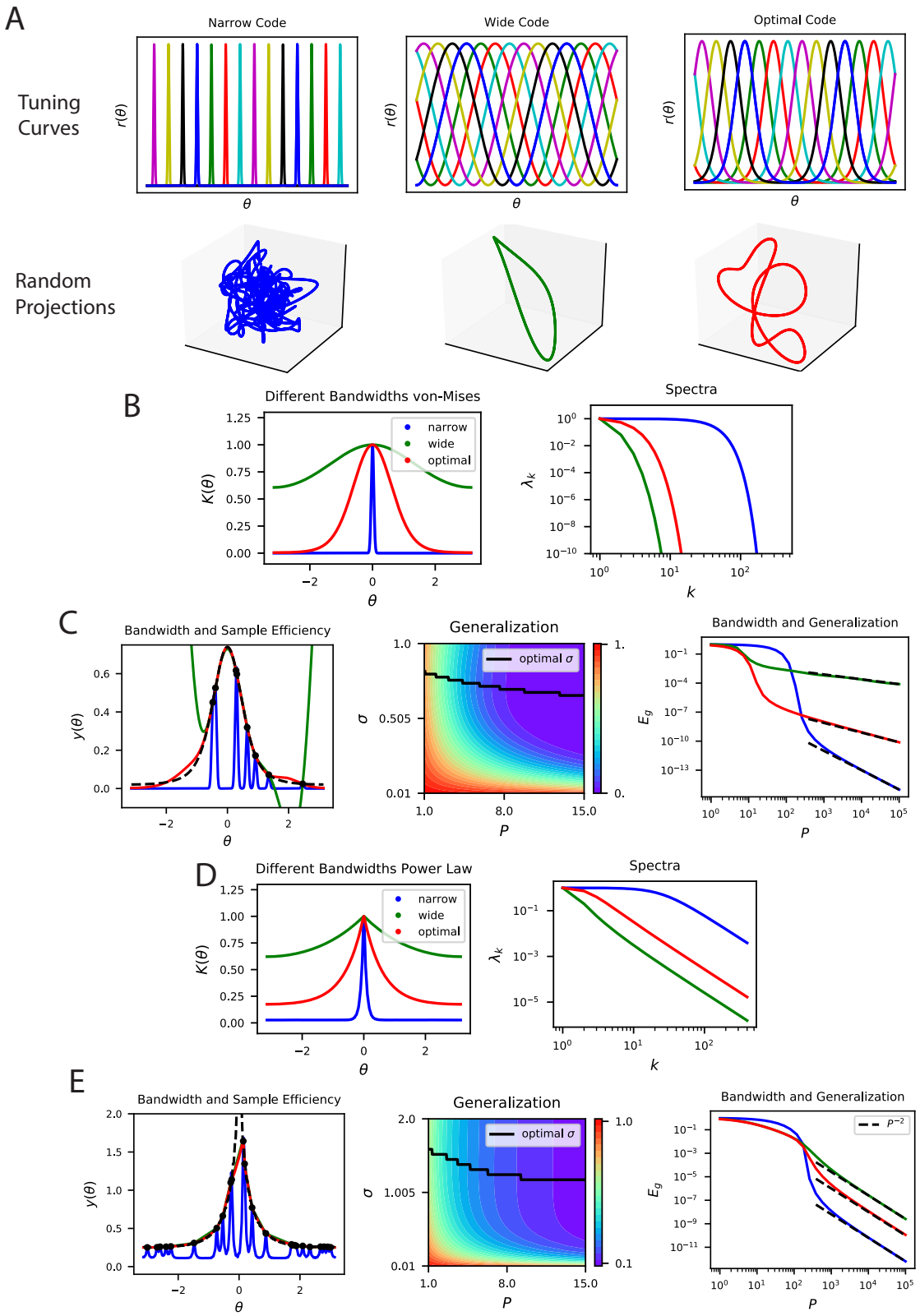

Figure 6: Caption on next page

Figure 6: The top eigensystem of a code determines its low-P generalization error. **A** A periodic variable is coded by a population of neurons with tuning curves of different widths (top). Narrow, wide and optimal refers to the example in C. These codes are all smooth (infinitely differentiable) but have very different feature space representations of the stimulus variable θ, as random projections reveal (below). **B** (left) The population codes in the above figure induce von Mises kernels $K(\theta) \propto e^{\cos(\theta)/\sigma^2}$ with different bandwidths σ. (right) Eigenvalues of the three kernels. **C** (left) As an example learning task, we consider estimating a "bump" target function. The optimal kernel (red, chosen as optimal bandwidth for $P = 10$) achieves a better generalization error than either the wide (green) or narrow (blue) kernels. (middle) A contour plot shows generalization error for varying bandwidth σ and sample size P. (right) The large P generalization error scales in a power law. Same color code as B and C-left. **D, E** The same experiment is performed with a Laplace kernel on a non-differentiable target function. The top eigenvalues are modified by changing the bandwidth, but the asymptotic power law scaling is preserved. Generalization at low P is shown in the contour plot while the large P scaling is provided in the generalization. In E-left and E-right, color code is the same as D.

Discussion

In this work, we examined how the structure of neural population codes shape inductive biases for learning. In computer science literature, "no-free lunch" theorems established that generalization is impossible without the right biases [6, 7, 8, 5]. Exploration of inductive bias and sample efficiency in the context of reinforcement learning [48], natural language processing [49], and computer vision [50, 51, 52] has shown how utilization of good representations of sensory data can accelerate the learning of a variety of natural tasks.

Our work suggests sample efficiency as a general coding principle for neural populations, relating neural representations to the kinds of problems they are well suited to solve. Prior related work demonstrated the dependence of sample-efficient learning of a two-angle estimation task on the width of the individual neural tuning curves [53]. A sample-efficiency approach to population coding theory differs from the classical normative theories in this domain, efficient coding theories [20, 18, 19, 54, 55, 56, 21, 57], which postulate that populations of neurons optimize information content of their code subject to metabolic constraints or noise. While these theories emphasize different aspect of the code's information content (such as reduced redundancy, predictive power, or sparsity), they do not address sample efficiency demands on learning.

We showed that neural codes shape inductive bias by enabling and constraining the learned readouts to accurately estimate certain stimulus-response maps instead of others in the small sample size regime. We showed that under the biologically-plausible delta rule, the generalization performance is entirely dependent on the code's inner product kernel, and proposed the kernel as a determiner of inductive bias. In its finite dimensional form, the kernel is an example of a representational similarity matrix and is a commonly used tool to study neural representations [58, 59, 60, 61, 62, 63]. Our work elucidates a concrete link between this mathematical object and sample-efficient learning.

We derived an analytical expression for the generalization error as a function of sample-size under very general conditions, for an arbitrary population code and an arbitrary target stimulus-response map. This generality of our theory is a particular strength. The ingredients of our theory are the singular values, and the right singular vectors of the neural response matrix, which in turn define kernel eigenvalues and eigenvectors. In previous works, the singular value decomposition of the neural response matrix has been traditionally used for dimensional reduction, which is a useful

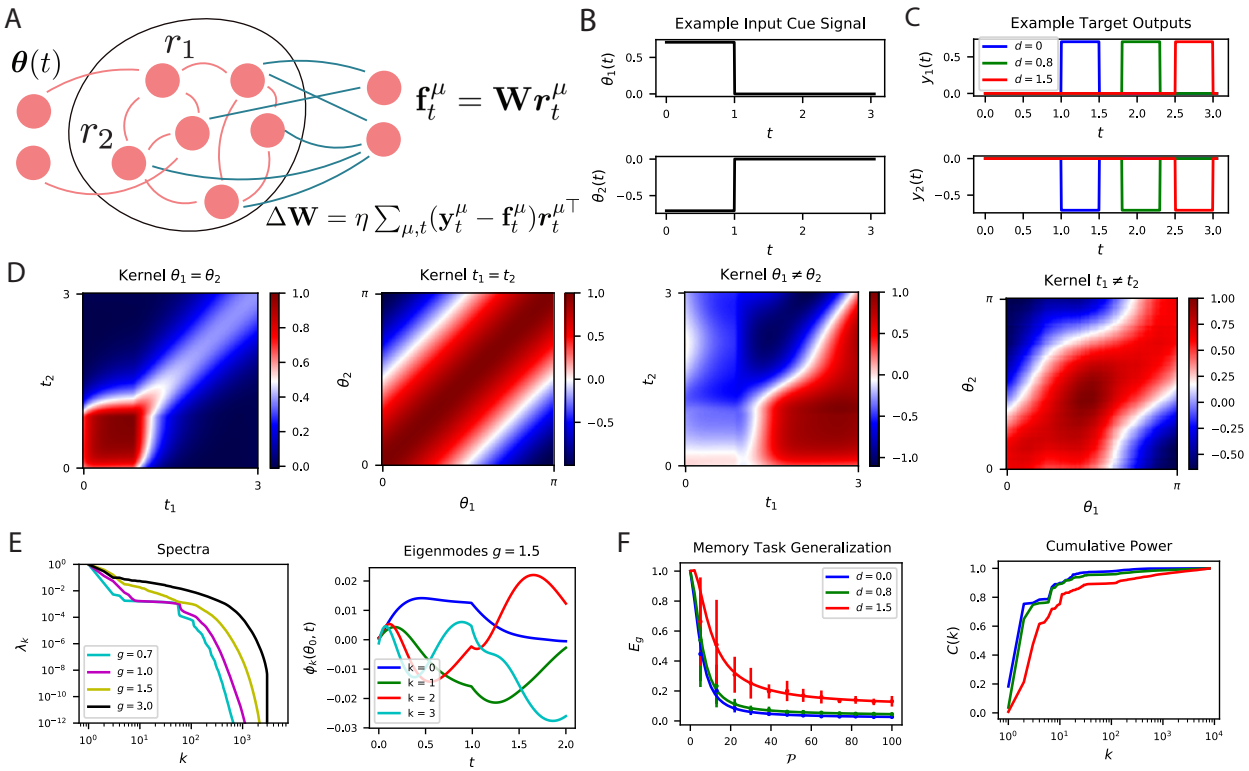

Figure 7: The performance of time-dependent codes when learning dynamical systems can be understood through spectral bias. **A** The readout weights (blue) of a time dependent code can be learned through a modified delta rule. **B** In a memory retrieval task, an input is presented to the network as a time series which terminates at $t = 1$. The sequences are generated by drawing an angle $\gamma^\mu \sim \text{Uniform}[0, 2\pi]$ and using input time-series $\boldsymbol{\theta}^\mu(t) = H(t)H(1-t)[\cos(\gamma^\mu), \sin(\gamma^\mu)]^\top$ where H is the Heaviside step function which lasts from $t = 0$ to $t = 1$. We show an example of the one of the variables in a input sequence. **C** The target functions for the memory retrieval task are step functions delayed by a time d. Specifically, $\mathbf{y}(\theta^\mu, t) = H(1.5 + d - t)H(t - d - 1)[\cos(\gamma^\mu), \sin(\gamma^\mu)]^\top$ which is nonzero for times $t \in [1 + d, 1.5 + d]$ where d represents a delay time. **D** The kernel $K_{\mu,\mu',t,t'}$ compares the code for two sequences at two distinct time points. We show the time dependent kernel for identical sequences (left) and the stimulus dependent kernel for equal time points (middle left) as well as for non-equal stimuli (middle right) and non-equal time (right). **E** The kernel can be diagonalized, and the eigenvalues λ_k determine the spectral bias of the reservoir computer (left). We see that higher gain g networks have higher dimensional representations. The first few "eigensystems" $\phi_k(\theta^\mu, t)$ are functions of time and cue angle (right). **F** The readout is trained to approximate a target function $y^\mu(t)$, which requires memory of the presented cue angle. The generalization error E_g is plotted for the three delays d and training sample size $\mathcal{P}$. The ordering of the generalization error matches the ordering of the $C(k)$ curves as expected.

tool in visualizing and elucidating the structure of neural representations [26, 27, 28, 29, 30, 31, 13, 10, 32, 33, 34].Such analysis is primarily concerned with the singular values and the left (as opposed to right) singular vectors of the neural response matrix (Figure 4). These two sides of the same singular value decomposition provide a conceptual link between the structure and function of neural codes.

Though the kernel is degenerate with respect to rotations of the code in the neural activity space, we demonstrated that the true V1 code has much lower metabolic cost than random codes with the same kernel, sugggesting that evolution and learning may be selecting neural codes with low average spike rates which preserve sample-efficiency demands for downstream learning tasks. The demand of metabolic efficiency is consistent with prior normative theories of coding [64, 21, 22, 65], however, our theory emphasizes the encoding needs for sample-efficient learning.

Our analysis elucidated two principles that define the inductive bias of the delta-rule readout of population codes. The first one is spectral bias: kernel eigenfunctions with large eigenvalues can be estimated using a smaller number of samples. Similar observations were make in the context of kernel regression [24, 25]. This finding defines a notion of "simplicity", and provides a solution to the question of what functions are easier to learn. The second principle is the code-task alignment: Target functions with most of their power in top kernel eigenfunctions can be estimated efficiently and are compatible with a code. The cumulative power distribution, $C(k)$ [25], provides a measure of this alignment.

We used these finding in both theoretical and experimental analysis of primary visual cortex (Figures 4, 5). We demonstrated a bias of neural populations towards low frequency orientation discrimination tasks. This bias can be modified by the neural thresholds and nonlinearities in nontrivial ways. The theoretical model of V1 as a mixture of simple and complex cells demonstrated how invariances, specifically the phase invariance of the complex cells, in the population code can facilitate learning some tasks involving phase invariant responses at the expense of performance on others.

Recent studies demonstrated hallmarks of redundancy and correlation in population responses [66, 28, 67, 34, 68, 69, 13]. These findings violate a generic prediction of efficient coding theories that responses of different neurons should be uncorrelated across input stimuli in high signal-to-noise regimes to reduce redundancy in the code and maximize information content [18, 19, 54, 55, 70, 71]. In our theory, the structured correlations of neural responses correspond to the decay in the spectrum of the kernel, and play a key role in biasing learned readouts towards simple functions.

In a related study, the asymptotic decay rate of the kernel's eigenspectrum was recently argued to be important for generalization [13]. Decay rate in the mouse visual cortex was found to be consistent with a high dimensional (power law) but smooth (differentiable) code, and smoothness was argued to be an enabler of generalization. We show that sample-efficient learning requires more than smoothness conditions in the form of asymptotic decay rates on the kernel's spectrum. The interplay between the stimulus distribution, target response and the code give rise to sample efficient learning. Because of spectral bias, the top eigenvalues govern the small sample size behavior. The tail of the spectrum becomes important for large sample sizes.

Our work focused on the effect of signal correlations to coding and inductive bias [72, 73]. Future analysis could study how signal and noise correlations interact to shape inductive bias and determine generalization.

Acknowledgements We thank C. Stringer, M. Pachitariu, M. Michaelos, N. Steinmetz, M. Carandini, and K. D.Harris for publicly sharing their datasets. We thank B. Ölveczky, C. Stringer, M. Pachitariu, K. Blum and J. Zavatone-Veth for comments on the manuscript.

Methods

Generating example codes (Figure 1)

The two codes in Figure 1 were constructed to produce two different kernels for $\theta \in S^1$:

$$K_1(\theta, \theta') = \exp(0.25\cos(\theta - \theta'))\ ,\ K_2(\theta, \theta') = \sum_{k=1}^{20} \cos(k(\theta - \theta')) \tag{8}$$

An infinite number of codes could generate either of these kernels. After diagonalizing the kernel into its eigenfunctions on a grid of 120 points, $\mathbf{K}_1 = \mathbf{\Phi}_1 \mathbf{\Lambda}_1 \mathbf{\Phi}_1^\top, \mathbf{K}_2 = \mathbf{\Phi}_2 \mathbf{\Lambda}_2 \mathbf{\Phi}_2^\top$, we used a random rotation matrix $\mathbf{Q} \in O(120)$ to generate a valid code

$$\mathbf{R}_1 = \mathbf{Q}\mathbf{\Lambda}_1^{1/2}\mathbf{\Phi}_1\ ,\ \mathbf{R}_2 = \mathbf{Q}\mathbf{\Lambda}_2^{1/2}\mathbf{\Phi}_2. \tag{9}$$

This construction guarantees that $\mathbf{R}_1^\top \mathbf{R}_1 = \mathbf{K}_1$ and $\mathbf{R}_2^\top \mathbf{R}_2 = \mathbf{K}_2$. We plot the tuning curves for the first three neurons. The target function in the first experiment is $y = \cos(\theta) - 0.6\cos(4\theta)$, while the second experiment used $y = \cos(6\theta) - \cos(8\theta)$.

Learning task and convergence of the delta-rule

Gradient descent training of readout weights $\mathbf{w}$ on a finite sample of size P converges to the kernel regression solution [74, 75, 76]. Let $\mathcal{D} = \{\boldsymbol{\theta}^\mu, y^\mu\}_{\mu=1}^P$ be the dataset with samples $\mathbf{x}^\mu$ and target values y^μ. We introduce a shorthand $\boldsymbol{r}^\mu = \boldsymbol{r}(\boldsymbol{\theta}^\mu)$ for convenience. The empirical loss we aim to minimize is a sum of the squared losses of each data point in the training set

$$\mathcal{L}(\mathbf{w}) = \frac{1}{2}\sum_{\mu=1}^{P}(\boldsymbol{r}^\mu \cdot \mathbf{w} - y^\mu)^2. \tag{10}$$

Performing gradient descent updates generates the following weight update

$$\mathbf{w}_{t+1} = \mathbf{w}_t - \eta\frac{\partial \mathcal{L}}{\partial \mathbf{w}} = \mathbf{w}_t - \eta\sum_{\mu=1}^{P}\boldsymbol{r}^\mu(\boldsymbol{r}^\mu \cdot \mathbf{w}_t - y^\mu), \tag{11}$$

which is merely the delta rule that we discussed in the main text [77, 78]. The dynamics for this rule can be analyzed efficiently through the singular value decomposition of the P-sample response matrix $\mathbf{R} = [\boldsymbol{r}^1, \boldsymbol{r}^2, ..., \boldsymbol{r}^P] \in \mathbb{R}^{N\times p}$. The singular value decomposition of $\mathbf{R} = \sum_{k=1}^P \sqrt{\hat{\lambda}_k}\hat{\mathbf{u}}_k\hat{\boldsymbol{\phi}}_k$ allows us to simplify the dynamics and identify the unique fixed point of the delta-rule. The singular value decomposition of this random sub-sample matrix $\mathbf{R}$ is different from the population singular value decomposition which is the solution to an integral eigenvalue problem (discussed in the next section). To clarify this, we use the "hat" $\hat{\lambda}_k, \hat{\mathbf{u}}_k, \hat{\boldsymbol{\phi}}_k$ to denote the singular components of the empirical matrix $\mathbf{R}$. We can expand $\mathbf{w}$ and $\mathbf{y}$ in the basis defined by $\hat{\mathbf{u}}_k$ and $\hat{\boldsymbol{\phi}}_k$ respectively so that $\mathbf{w}_t = \sum_k a_k^t \hat{\mathbf{u}}_k$ and $\mathbf{y} = \sum_k b_k \hat{\boldsymbol{\phi}}_k$. In this basis, the delta rule dynamics decouple

$$a_k^{t+1} = (1 - \eta\hat{\lambda}_k)a_k^t + \eta\sqrt{\hat{\lambda}_k}b_k\ ,\ k = 1, ..., p. \tag{12}$$

If we initialize the weights at the origin $\mathbf{w} = 0$, then we can solve these dynamics in closed form

$$a_k^t = \eta\sqrt{\hat{\lambda}_k}b_k\sum_{t'=0}^{t-1}(1 - \eta\lambda_k)^{t'} \tag{13}$$

which has the limit

$$\lim_{t\to\infty} a_k^t = \eta\sqrt{\hat{\lambda}_k} b_k \sum_{t'=0}^{\infty} (1-\eta\hat{\lambda}_k)^{t'} = \frac{\eta\sqrt{\hat{\lambda}_k} b_k}{1-(1-\eta\lambda_k)} = \frac{b_k}{\sqrt{\lambda_k}}, \tag{14}$$

where we used the fact of convergence of a geometric series $\sum_{k=0}^{\infty} z^k = \frac{1}{1-z}$ provided that $|z| < 1$. The equivalent condition for convergence in this case is that $|1-\eta\hat{\lambda}_k| < 1$ which implies $\eta < 2/\hat{\lambda}_k$ for all k. These dynamics converge to a unique fixed point $\mathbf{w}^*$

$$\mathbf{w}^* = \sum_{k:\hat{\lambda}_k>0} \frac{b_k}{\sqrt{\hat{\lambda}_k}} \hat{\mathbf{u}}_k, \tag{15}$$

where the sum runs over the modes k with nonzero eigenvalues $\hat{\lambda}_k > 0$. This solution is the minimum norm solution to the linear system $\mathbf{R}\mathbf{R}^\top \mathbf{w} = \mathbf{R}\mathbf{y}$ which can be written as $\mathbf{w}^* = \mathbf{R}\mathbf{K}^+\mathbf{y}$ where $\mathbf{K}^+$ is the Moore Penrose pseudo-inverse of the kernel gram matrix $\mathbf{K} = \mathbf{R}^\top\mathbf{R} \in \mathbb{R}^{P\times P}$ which is explicitly given by

$$\mathbf{K}^+ = \sum_{k:\hat{\lambda}_k>0} \frac{\hat{\boldsymbol{\phi}}_k \hat{\boldsymbol{\phi}}_k^\top}{\hat{\lambda}_k}. \tag{16}$$

Using these weights $\mathbf{w}^*$, we can calculate the learned function at a test point, we find

$$f(\boldsymbol{\theta}) = \boldsymbol{r}(\boldsymbol{\theta}) \cdot \mathbf{w}^* = \mathbf{k}(\boldsymbol{\theta}) \cdot \mathbf{K}^+\mathbf{y}, \tag{17}$$

where $k_\mu(\boldsymbol{\theta}) = K(\boldsymbol{\theta}, \boldsymbol{\theta}^\mu)$. This solution is known as the *kernel regression* solution for dataset $\mathcal{D}$ and kernel $K(\boldsymbol{\theta}, \boldsymbol{\theta}') = \boldsymbol{r}(\boldsymbol{\theta}) \cdot \boldsymbol{r}(\boldsymbol{\theta}')/N$.

Theory of Generalization

Recent work has established analytic results that predict the average case generalization error for kernel regression

$$E_g = \langle E_g(\mathcal{D}) \rangle_{\mathcal{D}} = \left\langle (f(\boldsymbol{\theta}, \mathcal{D}) - y(\boldsymbol{\theta}))^2 \right\rangle_{\boldsymbol{\theta},\mathcal{D}} \tag{18}$$

where $E_g(\mathcal{D}) = \left\langle (f(\boldsymbol{\theta}, \mathcal{D}) - y(\boldsymbol{\theta}))^2 \right\rangle_{\boldsymbol{\theta}}$ is the generalization error for a certain sample $\mathcal{D}$ of size P and $f(\boldsymbol{\theta}, \mathcal{D})$ is the kernel regression solution for $\mathcal{D}$ [24, 25]. The typical or average case error E_g is obtained by averaging over all possible datasets of size P. This average case generalization error is determined solely by the decomposition of the target function $y(\mathbf{x})$ along the eigenbasis of the kernel and the eigenspectrum of the kernel. This continuous diagonalization again takes the form

$$\int p(\boldsymbol{\theta}) K(\boldsymbol{\theta}, \boldsymbol{\theta}') \phi_k(\boldsymbol{\theta}) d\boldsymbol{\theta} = \lambda_k \phi_k(\boldsymbol{\theta}') \tag{19}$$

Since the eigenfunctions form a complete set of square integrable functions [17], we expand both the target function $y(\boldsymbol{\theta})$ and the learned function $f(\boldsymbol{\theta})$ in this basis

$$y(\boldsymbol{\theta}) = \sum_k v_k \phi_k(\boldsymbol{\theta}) \ , \ f(\boldsymbol{\theta}) = \sum_k w_k \phi_k(\boldsymbol{\theta}) \tag{20}$$

Due to the orthonormality of the kernel eigenfunctions $\{\phi_k\}$, the generalization error for any set of coefficients $\mathbf{w}$ is

$$E_g(\mathbf{w}) = \left\langle (y(\boldsymbol{\theta}) - f(\boldsymbol{\theta}))^2 \right\rangle_{\boldsymbol{\theta}} = \sum_k (w_k - v_k)^2 = ||\mathbf{w} - \mathbf{v}||^2 \tag{21}$$

We now introduce training error, or empirical loss, which depends on the disorder in the dataset $\mathcal{D} = \{(\boldsymbol{\theta}^\mu, y^\mu)\}_{\mu=1}^P$

$$H(\mathbf{w}, \mathcal{D}) = \sum_\mu (\mathbf{w} \cdot \boldsymbol{\phi}(\boldsymbol{\theta}^\mu) - \mathbf{v} \cdot \boldsymbol{\phi}(\boldsymbol{\theta}^\mu))^2 + \lambda \sum_k \frac{w_k^2}{\lambda_k} \tag{22}$$

It is straightforward to verify that the optimal $\mathbf{w}^*$ which minimizes $H(\mathbf{w}, \mathcal{D})$ is the kernel regression solution for kernel with eigenvalues $\{\lambda_k\}$ when $\lambda \to 0$. The optimal weights $\mathbf{w}$ can be identified through the first order condition $\nabla H(\mathbf{w}, \mathcal{D}) = 0$ which gives

$$\mathbf{w}^* = (\boldsymbol{\Phi}\boldsymbol{\Phi}^\top + \lambda \boldsymbol{\Lambda}^{-1})^{-1} \boldsymbol{\Phi}\boldsymbol{\Phi}^\top \mathbf{v} = \mathbf{v} - \lambda (\boldsymbol{\Phi}\boldsymbol{\Phi}^\top + \lambda \boldsymbol{\Lambda}^{-1})^{-1} \boldsymbol{\Lambda}^{-1} \mathbf{v}, \tag{23}$$

where $\Phi_{k,\mu} = \phi_k(\mathbf{x}^\mu)$ are the eigenfunctions evaluated on the training data and $\Lambda_{k,\ell} = \delta_{k,\ell}\lambda_k$ is a a diagonal matrix containing the kernel eigenvalues. The generalization error for this optimal solution is

$$E_g(\mathcal{D}) = ||\mathbf{w}^* - \mathbf{v}||^2 = \mathbf{v}^\top \boldsymbol{\Lambda}^{-1} \mathbf{G}(\mathcal{D})^2 \boldsymbol{\Lambda}^{-1} \mathbf{v} \ , \ \mathbf{G}(\mathcal{D}) = \left(\frac{1}{\lambda} \boldsymbol{\Phi}\boldsymbol{\Phi}^\top + \boldsymbol{\Lambda}^{-1} \right)^{-1}. \tag{24}$$

We note that the dependence on the randomly sampled dataset $\mathcal{D}$ only appears through the matrix $\mathbf{G}(\mathcal{D})$. Thus to compute the *typical* generalization error we need to average over this matrix $\langle \mathbf{G}(\mathcal{D}) \rangle_{\mathcal{D}}$. There are multiple strategies to perform such an average and we will study one here based on a partial differential equation which was introduced in [79, 80] and studied further in [24, 25]. We describe in detail how such an average can be performed in the SI. After this computation, we find that the generalization error can be written as

$$E_g = \frac{\kappa^2}{1-\gamma} \sum_k \frac{v_k^2}{(\lambda_k p + \kappa)^2} \ , \ \kappa = \lambda + \kappa \sum_k \frac{\lambda_k}{\lambda_k p + \kappa}, \tag{25}$$

where $\gamma = P \sum_k \frac{\lambda_k^2}{(\lambda_k p + \kappa)^2}$, giving the desired result. Taking $\lambda \to 0$ gives the generalization error of the minimum norm interpolant, which desribes the generalization error of the solution in (17). This result was recently reproduced using the replica method from statistical mechanics [24, 25]. Through implicit differentiation it is straightforward to verify that the ordering of the mode errors $E_k = \frac{\kappa^2}{1-\gamma} (\lambda_k p + \kappa)^{-2}$ matches the ordering of the eigenvalues [25].

V1 Model

A Simple Feedforward Model of V1

We consider a simplified but instructive model of the V1 population code as a linear-nonlinear map from photoreceptor responses through Gabor filters and then nonlinearity [40, 81, 82]. Let $\mathbf{x} \in \mathbb{R}^2$ represent the two-dimensional retinotopic position of photoreceptors. The firing rates of the photoreceptor at position $\mathbf{x}$ to a static grating stimulus oriented at angle θ is

$$h(\mathbf{x}, \theta) = \cos(\mathbf{k}(\theta) \cdot \mathbf{x}) \ , \ \mathbf{k} = \begin{bmatrix} \cos(\theta) \\ \sin(\theta) \end{bmatrix} \in \mathbb{R}^2 \ , \ \theta \in [0, 2\pi] \tag{26}$$

We model each V1 neuron's receptive field is a Gabor filter of the receptor responses $h(\mathbf{x}, \theta)$. The i-th V1 neuron has preferred wavevector $\mathbf{k}_i$, generating the following set of weights between photoreceptors and the i-th V1 neuron

$$w(\mathbf{x}, \theta_i) = \frac{\sigma^2}{2\pi} e^{-\frac{\sigma^2}{2}|\mathbf{x}|^2} \cos(\mathbf{k}(\theta_i) \cdot \mathbf{x}). \tag{27}$$

The V1 population code is obtained by filtering the photoreceptor responses. By approximating the resulting sum over all retinal photoreceptors with an integral, we find the response of neuron i to grating stimulus with wavenumber $\mathbf{k}$ is

$$r_i(\theta) = \mathbf{w}(\theta_i) \cdot \mathbf{h}(\theta) = \int w(\mathbf{x}, \theta_i) h(\mathbf{x}, \theta) d\mathbf{x} = \frac{\sigma^2}{2\pi} \int e^{-\frac{\sigma^2}{2}|\mathbf{x}|^2} \cos(\mathbf{k}(\theta_i) \cdot \mathbf{x}) \cos(\mathbf{k}(\theta) \cdot \mathbf{x}) d\mathbf{x}. \tag{28}$$

Direct integration yields the following tuning curve for neuron i

$$r_i(\theta, \phi) = \frac{1}{2} e^{-\frac{1}{2\sigma^2}|\mathbf{k}+\mathbf{k}_i|^2} + \frac{1}{2} e^{-\frac{1}{2\sigma^2}|\mathbf{k}-\mathbf{k}_i|^2}. \tag{29}$$

Let $p(\mathbf{k}_i)$ represent the distribution of the preferred wavevectors and $p(\phi_i)$ the distribution of preferred phases. The trial averaged code has the following inner product kernel $K(\theta, \theta')$

$$K(\theta, \theta') = \int p(\mathbf{k}_i) r_i(\theta) r_i(\theta') d\mathbf{k}_i. \tag{30}$$

To allow computation, we will assume a specific distribution of preferred wavevectors as isotropic Gaussian $\mathbf{k}_i \sim \mathcal{N}(0, \tau^2 \mathbf{I})$ with different variances. Under this assumption, the kernel has the following form

$$K(\theta, \theta') = \frac{\cosh(\beta \cos(\theta - \theta'))}{\cosh(\beta)}, \tag{31}$$

where $\beta = \frac{\tau^2}{\sigma^2(\sigma^2+2\tau^2)}$ and the kernel is normalized to have maximum value of 1. Note that this normalization of the kernel is completely legitimate since merely rescales each eigenvalue by a constant and does not change the learning curves.

Since the kernel only depends on the difference between angles $\theta - \theta'$, it is said to posess translation invariance. Such translation invariant kernels admit a Mercer decomposition in terms of Fourier modes $K(\theta) = \sum_n \lambda_n \cos(n\theta)$ since the Fourier modes diagonalize shift invariant integral operators on $\mathbb{S}^1$. The Mercer Decomposition of this kernel has the form

$$K(\theta) = \frac{1}{\cosh(\beta)} \sum_{n=0}^{\infty} [I_n(\beta) + I_n(-\beta)] \cos(n\theta), \tag{32}$$

where I_n are the Bessel functions of the first kind. These Bessel functions evaluated at β are the kernel eigenvalues in this setting since it matches the form of the Mercer expansion for $\mathbb{S}^1$. For large n the eigenvalues decay like

$$I_n(\beta) \sim \frac{\beta^n}{2^n n!} \ , \ n \to \infty. \tag{33}$$

As expected this spectrum decays more rapidly than any power of n due to the infinite differentiability of the tuning curves. Since $I_n(\beta)$ decays rapidly with n, we find that this Gabor code has an inductive bias that favors low frequency functions of orientation θ.

Nonlinear Simple Cells

Introducing nonlinear functions ψ that map input currents into the V1 population into firing rates, we can obtain a non-linear kernel $K_\psi(\theta)$ which has the following definition

$$K_\psi(\mathbf{k}, \mathbf{k}') = \int p(\mathbf{k}_i) \psi(\mathbf{w}_i \cdot \mathbf{h}(\mathbf{k})) \psi(\mathbf{w}_i \cdot \mathbf{h}(\mathbf{k}')) d\mathbf{k}_i. \tag{34}$$

In this setting, it is convenient to restrict $\mathbf{k}_i, \mathbf{k}, \mathbf{k}' \in \mathbb{S}^1$ and assume that the preferred wavevectors $\mathbf{k}_i$ are uniformly distributed over the circle. In this case, it suffices to identify a decomposition of the composed function $\psi(\mathbf{w}_i \cdot \mathbf{h}(\theta))$ in the basis of Chebyshev polynomials $T_n(z)$ which satisfy $T_n(\cos(\theta)) = \cos(n\theta)$

$$a_n(\psi) = \frac{1}{2\pi} \int_0^{2\pi} \psi\left(e^{-\frac{1}{\sigma^2}} \cosh\left(\frac{1}{\sigma^2}\cos(\theta)\right)\right) \cos(n\theta) d\theta \tag{35}$$

$$= \frac{1}{2\pi} \int_{-1}^{1} \frac{1}{\sqrt{1-z^2}} \psi\left(e^{-\frac{1}{\sigma^2}} \cosh\left(z/\sigma^2\right)\right) T_n(z) dz, \tag{36}$$

which can be computed efficiently with an appropriate quadrature scheme. Once the coefficients a_n are determined, we can compute the kernel by first letting θ_i to be the angle between $\mathbf{k}$ and $\mathbf{k}_i$ and letting θ be the angle between $\mathbf{k}$ and $\mathbf{k}'$

$$K_\psi(\theta) = \int_0^{2\pi} \frac{d\theta_i}{2\pi} \sum_{n,n'} a_n(\psi) a_{n'}(\psi) T_n(\cos(\theta_i)) T_{n'}(\cos(\theta_i + \theta)) d\theta_i \tag{37}$$

$$= \sum_{n,n'} a_n(\psi) a_{n'}(\psi) \frac{1}{2\pi} \int_0^{2\pi} \cos(n\theta_i) \left[\cos(n'\theta_i)\cos(n'\theta) + \sin(n'\theta_i)\sin(n'\theta)\right] d\theta_i \tag{38}$$

$$= \frac{1}{2} \sum_n a_n^2(\psi) \cos(n\theta). \tag{39}$$

Thus the kernel eigenvalues are $\lambda_n = \frac{1}{2} a_n^2(\psi)$. Activation functions that encourage sparsity have slower eigenvalue decays. If the nonlinear f-I activation function has the form $\psi_{q,t}(z) = \max\{0, z-t\}^q$, then the spectrum decays like $\lambda_n \sim n^{-2q-2}$. A simple argument justifies this scaling: if the function $\psi(e^{-\sigma^2}\cosh(\sigma^2 z))$ is only $q-1$ times differentiable then $a_n n^q \sim n^{-1}$ since $\sum_n a_n n^q$ must diverge. Therefore $\lambda_n = a_n^2 \sim n^{-2q-2}$.

Phase Variation, Complex Cells and Invariance

We can consider a slightly more complicated model where Gabors and stimuli have phase shifts

$$h(\mathbf{x}, \theta, \phi) = \cos(\mathbf{k}(\theta) \cdot \mathbf{x} - \phi) \ , \ w(\mathbf{x}, \theta_i, \phi_i) = \frac{\sigma^2}{2\pi} e^{-\frac{\sigma^2}{2}|\mathbf{x}|^2} \cos(\mathbf{k}_i \cdot \mathbf{x} - \phi_i). \tag{40}$$

The simple cells are generated by nonlinearity

$$r_i(\theta, \phi) = \psi(\mathbf{w}(\theta_i, \phi_i) \cdot \mathbf{h}(\theta, \phi)). \tag{41}$$

The input currents into the simple V1 cells can be computed exactly

$$\mathbf{w}(\theta_i, \phi_i) \cdot \mathbf{h}(\theta, \phi) = \langle \cos(\mathbf{k}_i \cdot \mathbf{x} - \phi_i) \cos(\mathbf{k} \cdot \mathbf{x} - \phi) \rangle_{\mathbf{x} \sim \mathcal{N}(0, \sigma^2 \mathbf{I})} . \tag{42}$$

$$= \frac{1}{2} \cos(\phi + \phi_i) e^{-\frac{1}{2\sigma^2}|\mathbf{k}+\mathbf{k}_i|^2} + \frac{1}{2} \cos(\phi - \phi_i) e^{-\frac{1}{2\sigma^2}|\mathbf{k}-\mathbf{k}_i|^2} \tag{43}$$

When $|\mathbf{k}| = |\mathbf{k}_i| = 1$, the simple cell tuning curves $r_i = \psi(\mathbf{w}_i \cdot \mathbf{h})$ only depend on $\cos(\theta - \theta_i)$ and ϕ, allowing a Fourier decomposition

$$r_i(\theta, \phi) = \sum_n a_n(\phi, \phi_i) \cos(n(\theta - \theta_i))). \tag{44}$$

The simple cell kernel K_s, therefore decomposes into Fourier modes over θ

$$K_s(\theta, \theta', \phi, \phi') = \sum_n b_n(\phi, \phi') \cos(n(\theta - \theta')). \tag{45}$$

where $b_n(\phi, \phi') = \langle a_n(\phi, \phi_i) a_n(\phi', \phi_i) \rangle_{\phi_i}$. It therefore suffices to solve the infinite sequence of integral eigenvalue problems over ϕ

$$\frac{1}{2\pi} \int_0^{2\pi} b_n(\phi, \phi') v_{n,k}(\phi) d\phi = \lambda_{n,k} v_{n,k}(\phi') \implies K_s(\theta, \theta', \phi, \phi') = \sum_{n,k} \lambda_{n,k} \cos(n(\theta - \theta')) v_{n,k}(\phi) v_{n,k}(\phi') \tag{46}$$

With this choice it is straightforward to verify that the kernel eigenfunctions are $v_{n,k}(\theta, \phi) = e^{in\theta} v_{n,k}(\phi)$ with corresponding eigenvalue $\lambda_{n,k}$. Since b_n is not translation invariant in $\phi - \phi'$, the eigenfunctions $v_{n,k}$ are not necessarily Fourier modes. These eigenvalue problems for b_n must be solved numerically when using arbitrary nonlinearity ψ. The top eigenfunctions of the simple cell kernel depend heavily on the phase of the two grating stimuli ϕ. Thus, a pure orientation discrimination task which is independent of phase requires a large number of samples to learn with the simple cell population.

Complex Cells Populations are Phase Invariant

V1 also contains complex cells which possess invariance to the phase ϕ of the stimulus. Again using Gabor filters

$$w(\mathbf{x}, \theta_i, \phi_i) = \frac{\sigma^2}{2\pi} e^{-\frac{\sigma^2}{2}|\mathbf{x}|^2} \cos(\mathbf{k}(\theta_i) \cdot \mathbf{x} - \phi_i), \tag{47}$$

we model the complex cell responses with a quadratic nonlinearity and sum over two squared filters which are phase shifted by $\pi/2$

$$r_i(\theta, \phi) = (\mathbf{w}(\theta_i, \phi_i) \cdot \mathbf{h}(\theta, \phi))^2 + (\mathbf{w}(\theta_i, \phi_i - \pi/2) \cdot \mathbf{h}(\theta, \phi))^2 = \frac{1}{4} e^{-\frac{1}{\sigma^2}|\mathbf{k}+\mathbf{k}_i|^2} + \frac{1}{4} e^{-\frac{1}{\sigma^2}|\mathbf{k}-\mathbf{k}_i|^2} + \frac{1}{2} e^{-\sigma^2} \cos(2\phi_i), \tag{48}$$

which we see is independent of the phase ϕ of the grating stimulus. Integrating over the set of possible Gabor filters $(\mathbf{k}_i, \phi_i)$ again gives the following kernel for the complex cells

$$K_c(\theta) = \frac{1}{\cosh(2\beta)} \cosh(2\beta \cos(\theta)). \tag{49}$$

Remarkably, this kernel is independent of the phase ϕ of the grating stimulus. Thus, complex cell populations possess good inductive bias for vision tasks where the target function only depends on the orientation of the stimulus rather than it's phase. In reality, V1 is a mixture of simple and complex cells. Let $s \in [0, 1]$ represent the relative proportion of neurons which are simple cells and $(1 - s)$ the relative proportion of complex cells. The kernel for the mixed V1 population is given by a simple convex combination of the simple and complex cell kernels

$$K_{V1}(\theta, \theta', \phi, \phi') = \frac{1}{N} \sum_{i=1}^{N} r_i(\theta, \phi) r_i(\theta', \phi') \to \left\langle r(\theta, \phi, c) r(\theta', \phi', n) \right\rangle_{n \sim p_{V1}(n)} \tag{50}$$

$$= s \left\langle r(\theta, \phi, n) r(\theta', \phi', n) \right\rangle_{n \sim p_s(n)} + (1 - s) \left\langle r(\theta, \phi, n) r(\theta', \phi', n) \right\rangle_{n \sim p_c(n)} \tag{51}$$

$$= s K_s(\theta, \theta', \phi, \phi') + (1 - s) K_c(\theta, \theta'), \tag{52}$$

where n denotes neuron type (simple vs complex, tuning etc) and $P_{V1}(n), p_s(n), p_c(n)$ are probability distributions over the V1 neuron identities, the simple cell identities and the complex cell identities respectively. Increasing s increases the phase dependence of the code by giving greater weight to the simple cell population. Decreasing s gives weight to the complex cell population, encouraging phase invariance of readouts.

Time-Dependent Neural Codes

In this setting, the population code $\boldsymbol{r}(\{\boldsymbol{\theta}(t)\}, t)$ is a function of an input stimulus sequence $\boldsymbol{\theta}(t)$ and time t. In general the neural code $\boldsymbol{r}$ at time t can depend on the entire history of the stimulus input $\boldsymbol{\theta}(t')$ for $t' \leq t$, as is the case for recurrent neural networks. We denote dependence of a function g on $\boldsymbol{\theta}(t)$ in this causal manner with the notation $g(\{\boldsymbol{\theta}\}, t)$. In a learning task, a set of readout weights $\mathbf{w}$ are chosen so that a downstream linear readout $f(\{\boldsymbol{\theta}\}, t) = \mathbf{w} \cdot \boldsymbol{r}(\{\boldsymbol{\theta}\}, t)$ approximates a target sequence $y(\{\boldsymbol{\theta}\}, t)$ which maps input stimulus sequences to output scalar sequences. The quantity of interest is the generalization E_g, which in this case is an average over both input sequences and time (Methods), $E_g = \left\langle (y(\{\boldsymbol{\theta}\}, t) - f(\{\boldsymbol{\theta}\}, t))^2 \right\rangle_{\boldsymbol{\theta}(t), t}$. The average is computed over a distribution of input stimulus sequences $p(\boldsymbol{\theta}(t))$. To train the readout, $\mathbf{w}$, the network is given a sample of P stimulus sequences $\boldsymbol{\theta}^\mu(t), \mu = 1, ..., P$. For the μ-th training input sequence, the target system y is evaluated at a set of discrete time points $\mathcal{T}_\mu = \{t_1, t_2, ..., t_{|\mathcal{T}_\mu|}\}$ giving a collection of target values $\{y_t^\mu\}_{t \in \mathcal{T}_\mu}$ and a total dataset of size $\mathcal{P} = \sum_{\mu=1}^{P} |\mathcal{T}_\mu|$. The *average case generalization* computes a further average of the generalization error E_g over randomly sampled datasets of size $\mathcal{P}$.

Learning is again achieved through iterated weight updates with delta-rule form, but now have contributions from both sequence index and time $\Delta \mathbf{w} = \eta \sum_\mu \sum_{t \in \mathcal{T}_\mu} \boldsymbol{r}_t^\mu (y_t^\mu - f_t^\mu)$. As before, optimization of the readout weights is equivalent to kernel regression with a kernel that computes inner products of neural population vectors at different times t, t' for different input sequences $\{\boldsymbol{\theta}\}, \{\boldsymbol{\theta}'\}$: $K(\{\boldsymbol{\theta}\}, \{\boldsymbol{\theta}'\}, t, t') = \frac{1}{N} \boldsymbol{r}(\{\boldsymbol{\theta}\}, t) \cdot \boldsymbol{r}(\{\boldsymbol{\theta}\}, t')$. This kernel depends on details of the time varying population code including its recurrent intrinsic dynamics as well as its encoding of the time-varying input stimuli. The optimization problem and delta rule described above converge to the kernel regression solution for kernel gram matrix $K_{t,t'}^{\mu,\mu'} = \frac{1}{N} \boldsymbol{r}_t^\mu \cdot \boldsymbol{r}_{t'}^{\mu'}$ [42, 43, 44]. The learned function has the form $f(\{\boldsymbol{\theta}\}, t) = \sum_{\mu, t' \in \mathcal{T}_\mu} \alpha_t^\mu K(\{\boldsymbol{\theta}\}, \{\boldsymbol{\theta}\}^\mu, t, t')$, where $\boldsymbol{\alpha} = \mathbf{K}^+ \mathbf{y}$ for kernel gram matrix $\mathbf{K} \in \mathbb{R}^{\mathcal{P} \times \mathcal{P}}$ which is computed for the entire set of training sequences, and the vector $\mathbf{y} \in \mathbb{R}^{\mathcal{P}}$ is the vector containing the desired target outputs for each sequence. Assuming a probability distribution over sequences $\boldsymbol{\theta}(t)$, the kernel can be diagonalized with orthonormal eigenfunctions $\phi_k(\{\boldsymbol{\theta}\}, t)$. Our theory carries over from the static case: kernels whose top eigenfunctions have high alignment with the target dynamical system $y(\{\boldsymbol{\theta}\}, t)$ will achieve the best average case generalization performance.

RNN Experiment

For the simulations in Figure 7 we integrated a rate based recurrent network model with $N = 6000$ neurons, time constant $\tau = 0.05$ and gain $g = 1.5$. The recurrent weights are drawn from a standard normal $W_{ij}^{\boldsymbol{r}} \sim \mathcal{N}(0, 1)$ as are the input encoding weights $W_{ij}^{\theta} \sim \mathcal{N}(0, 1)$. Each of the $P = 80$ randomly chosen angles γ^μ generates a trajectory over $T = 100$ equally spaced points in $t \in [0, 3]$. The two dimensional input sequence is simply $\boldsymbol{\theta}(t) = H(t)H(1-t) \left[\cos(\gamma^\mu), \sin(\gamma^\mu)\right]^\top \in \mathbb{R}^2$. The kernel gram matrix $\mathbf{K} \in \mathbb{R}^{PT \times PT}$ is computed by taking inner products of the time varying code at for different inputs γ^μ and at different times. Learning curves represent the generalization error obtained by randomly sampling $\mathcal{P}$ time points from the PT total time points generated in the simulation process and training readout weights $\mathbf{w}$ to convergence with gradient descent.

Data Analysis

Data source and processing

Mouse V1 neuron responses to orientation gratings were obtained from a publicly available dataset [10, 11]. Two-photon calcium microscopy fluorescence traces were deconvolved into spike trains and spikes were counted for each stimulus, as described in [10]. The presented grating angles were distributed uniformly over $[0, 2\pi]$ radians. Data pre-processing, which included z-scoring against the mean and standard deviation of null stimulus responses, utilized the provided code for this experiment, which also publicly available at `https://github.com/MouseLand/stringer-et-al-2019`. This preprocessing technique was used in all Figures in the paper. To reduce corruption of the estimated kernel from neural noise (trial-to-trial variability), we first trial average responses, binning the grating stimuli oriented at different angles θ into a collection of 100 bins over the interval from $[0, 2\pi]$ and averaging over all of the available responses from each bin. Since grating angles were sampled uniformly, there is a roughly even distribution of about 45 responses in each bin. After trial averaging, SVD was performed on the response matrix $\boldsymbol{R}$, generating the eigenspectrum and kernel eigenfunctions as illustrated in Figure 4. Figures 2, 3, 4, all used this data anytime responses to grating stimuli were mentioned.

In Figures 3C and 4D, the responses of mouse V1 neurons to ImageNet images were obtained from a different publicly available dataset [12, 13]. Again, spike counts were obtained from deconvolved and z-scored calcium fluorescence traces. Each of the images presented belongs to one of 15 relevant Imagenet categories, including the mice and bird categories displayed in 4D. The preprocessing code and image category information were obtained from the publicly available code base at `https://github.com/MouseLand/stringer-pachitariu-et-al-2018b`.

Generating alternative codes

In Figure 3, the randomly rotated codes are generated by sampling a matrix $\mathbf{Q}$ from the Haar measure on the set of N-by-N orthogonal matrices, and chosing a $\boldsymbol{\delta}$ by solving the following optimization problem:

$$\min_{\boldsymbol{\delta}\in\mathbb{R}^N} \sum_{i=1}^{N}\sum_{\mu=1}^{P} s_i^\mu, \qquad s.t. \quad \mathbf{s}^\mu = \mathbf{Q}\boldsymbol{r}(\boldsymbol{\theta}^\mu) + \boldsymbol{\delta}, \quad s_i^\mu \geq 0, \quad i = 1, \ldots, N, \quad \mu = 1, \ldots, P, \tag{53}$$

which minimizes the total spike count subject to the kernel and nonnegativity of firing rates. The solution to this problem is given by $\delta_i^* = -\min_{\mu=1,\ldots,P}[\mathbf{Q}\boldsymbol{r}(\boldsymbol{\theta}^\mu)]_i$.

Comparing Sparsity of Population Codes

To explore the metabolic cost among the set of codes with the same inductive biases, we estimate the distribution of average spike counts of codes with the same inner product kernel as the biological code. These codes are generated in the form $\boldsymbol{s}^\mu = \mathbf{Q}\boldsymbol{r}^\mu + \boldsymbol{\delta}$ where $\boldsymbol{\delta}$ solves the optimization problem

$$\min_{\boldsymbol{\delta}\in\mathbb{R}^N} \sum_{i,\mu} s_i^\mu \ , \ s.t. \ \boldsymbol{s}^\mu = \mathbf{Q}\boldsymbol{r}^\mu + \boldsymbol{\delta} \ , \ s_i^\mu \geq 0 \tag{54}$$

To quantify the distribution of such codes, we randomly sample $\mathbf{Q}$ from the Haar-measure on $O(N)$ and compute the optimal $\boldsymbol{\delta}$ as described above. This generates the aqua colored distribution in Figure 3 B and C.

We also attempt to characterize the most efficient code with the same inner product kernel

$$\min_{\mathbf{Q}\in O(N),\boldsymbol{\delta}} \sum_{i,\mu} s_i^\mu \ , \ s.t. \ \boldsymbol{s}^\mu = \mathbf{Q}\boldsymbol{r}^\mu + \boldsymbol{\delta} \ , \ s_i^\mu \geq 0. \tag{55}$$

Since this optimization problem is non-convex in $\mathbf{Q}$, there is no theoretical guarantee that minima are unique. Nonetheless, we attempt to optimize the code by starting $\mathbf{Q}$ at the identity matrix and conduct gradient descent in the tangent space $so(N)$. Such updates take the form

$$\mathbf{Q}_{t+1} = \exp(-\eta\nabla\mathcal{L})\mathbf{Q}_t \ , \ \nabla\mathcal{L} = \frac{\partial\mathcal{L}}{\partial\mathbf{Q}}\mathbf{Q}^\top - \mathbf{Q}\frac{\partial\mathcal{L}}{\partial\mathbf{Q}}^\top \tag{56}$$

where $\exp(\cdot)$ is the matrix exponential. To make the loss function differentiable, we incorporate the non-negativity constraint with a soft-minimum:

$$\mathcal{L} = \sum_{i\mu}\left(\mathbf{q}_i^\top\boldsymbol{r}^\mu - \text{softmin}_\nu(\mathbf{q}_i^\top\boldsymbol{r}^\nu,\beta)\right) \ , \quad \text{softmin}(a_1,a_2,...,a_P;\beta) = \frac{1}{Z}\sum_{\mu=1}^{P} a_\nu \exp(-\beta a_\nu), \tag{57}$$

where $Z = \sum_\nu \exp(-\beta a_\nu)$ is a normalizing constant and $\mathbf{Q} = [\mathbf{q}_1, ...\mathbf{q}_N]$. In the $\beta \to \infty$ limit, this cost function converges to the exact optimization problem with non-negativity constraint. Finite β, however, allows learning with gradient descent. Gradients are computed with automatic differentiation in JAX [83]. This optimization routine is run until convergence and the optimal value is plotted as dashed red lines labeled "optimal" in Figure 3.

We show that our result is robust to different pre-processing techniques and to imposing bounds on neural firing rates in Figure SI.1. To demonstrate that our result is not an artifact of z-scoring the deconvolved signals against the spontaneous baseline activity level, we also conduct the random rotation experiment on the raw deconvolved signals. In addition, we show that imposing realistic constraints on the upper bound of the each neuron's responses does not change our findings. We used a subset of $N = 100$ neurons and computed random rotations. However, we only accepted a code as valid if it's maximum value was less than some upper bound u_b. Subsets of $N = 100$ neurons in the biological code achieve maxima in the range between 3.2 and 4.7. We performed this experiment for $u_b \in \{3, 4, 5\}$ so that the artificial codes would have maxima that lie in the same range as the biological code.

Fitting a Gabor model to mouse V1 kernel

Under the assumption of translation symmetry in the kernel $K(\theta, \theta')$, we averaged the elements of the over rows of the empirical mouse V1 kernel [11]

$$K(\Delta) = \frac{1}{P}\sum_{\mu=1}^{P} K(\theta^\mu, \theta^\mu + \Delta) \tag{58}$$

where angular addition is taken mod π. This generates the black dots in Figure 5 B. We aimed to fit a threshold-power law nonlinearity of the form $\psi_{q,a}(z) = \max\{0, z-a\}^q$ to the kernel. Based on the Gabor model discussed above, we parameterized tuning curves as

$$r_{s,q,a}(\theta,\theta_i) = \psi_{q,a}\left(\frac{\cosh(s\cos(\theta-\theta_i))}{\cosh(s)}\right), \tag{59}$$

where θ_i is the preferred angle of the i-th neuron's tuning curve. Rather than attempting to perform a fit of $s, a, q, \{\theta_i\}_{i=1}^N$ of this form to the responses of each of the $\sim$ 20-k neurons, we instead simply

attempt to fit to the population kernel by optimizing over (s, a, q). However, we noticed that two of these variables s, a are constrained by the sparsity level of the code. If each neuron, on average, fires for only a fraction f of the uniformly sampled angles θ, then the following relationship holds between s and a

$$a = \frac{\cosh\left(s\cos\left(\frac{\pi}{2}f\right)\right)}{\cosh(s)}. \tag{60}$$

Calculation of the coding level f for the recorded responses allowed us to infer a from s during optimization. This reduced the free parameter set to (s, q). We then solve the following optimization problem

$$\min_{s,q} \left\langle \left(\hat{K}_{s,q}(\theta) - K(\theta)\right)^2 \right\rangle_\theta \ , \ \hat{K}_{s,q}(\theta) = \frac{1}{\pi}\int_{-\pi/2}^{\pi/2} r_{s,q}(\theta,\theta_i) r_{s,q}(0,\theta_i) d\theta_i, \tag{61}$$

where integration over θ_i is performed numerically. Using the Scipy Trust-Region constrained optimization routine, we found $(q, s, a) = (1.7, 5.0, 0.2)$ which we use as the fit parameters in Figure 5.

References

[1] Susan Carey and Elsa Bartlett. Acquiring a single new word. 1978.

[2] Matthew F. Peterson, Craig K. Abbey, and Miguel P. Eckstein. The surprisingly high human efficiency at learning to recognize faces. *Vision Research*, 49(3):301–314, 2009.

[3] Joshua B Tenenbaum, Charles Kemp, Thomas L Griffiths, and Noah D Goodman. How to grow a mind: Statistics, structure, and abstraction. *science*, 331(6022):1279–1285, 2011.

[4] Brenden M. Lake, Tomer D. Ullman, Joshua B. Tenenbaum, and Samuel J. Gershman. Building machines that learn and think like people. *Behavioral and Brain Sciences*, 40:e253, 2017.

[5] Fabian H Sinz, Xaq Pitkow, Jacob Reimer, Matthias Bethge, and Andreas S Tolias. Engineering a less artificial intelligence. *Neuron*, 103(6):967–979, 2019.

[6] David H Wolpert, William G Macready, et al. No free lunch theorems for search. Technical report, Technical Report SFI-TR-95-02-010, Santa Fe Institute, 1995.

[7] David H Wolpert. The lack of a priori distinctions between learning algorithms. *Neural computation*, 8(7):1341–1390, 1996.

[8] David H Wolpert and William G Macready. No free lunch theorems for optimization. *IEEE transactions on evolutionary computation*, 1(1):67–82, 1997.

[9] David Hume. *An enquiry concerning human understanding : a critical edition.* Hume, David, 1711-1776. Works. 1998. Clarendon Press ; Oxford University Press, Oxford : New York, 2000.

[10] C. Stringer, M. Michaelos, and M. Pachitariu. High precision coding in mouse visual cortex. *bioRxiv*, 2019.

[11] Marius Pachitariu, Michalis Michaelos, and Carsen Stringer. Recordings of 20,000 neurons from V1 in response to oriented stimuli. *10.25378/janelia.8279387.v3*, 11 2019.

[12] Carsen Stringer, Marius Pachitariu, Matteo Carandini, and Kenneth Harris. Recordings of 10,000 neurons in visual cortex in response to 2,800 natural images, Jul 2018.

[13] Carsen Stringer, Marius Pachitariu, Nicholas Steinmetz, Matteo Carandini, and Kenneth D. Harris. High-dimensional geometry of population responses in visual cortex. *Nature*, 2018.

[14] M. Radford Neal. Bayesian learning for neural networks. *PhD Thesis, University of Toronto Department of Computer Science*, 1994.

[15] Federico Girosi, Michael Jones, and Tomaso Poggio. Regularization theory and neural networks architectures. *Neural Computation*, 7(2):219–269, 1995.

[16] Bernhard Schölkopf, Ralf Herbrich, and Alex J. Smola. A generalized representer theorem. In *Proceedings of the 14th Annual Conference on Computational Learning Theory and and 5th European Conference on Computational Learning Theory*, COLT '01/EuroCOLT '01, page 416–426, Berlin, Heidelberg, 2001. Springer-Verlag.

[17] Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian Processes for Machine Learning (Adaptive Computation and Machine Learning)*. The MIT Press, 2005.

[18] H. Barlow. Possible principles underlying the transformation of sensory messages. *Sensory Communication, MIT Press*, 1961.

[19] J. J. Atick and A. N. Redlich. What does the retina know about natural scenes? *Neural Computation*, 4(2):196–210, March 1992.

[20] Fred Attneave. Some informational aspects of visual perception. *Psychological review*, 61(3):183, 1954.

[21] 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, 1997.

[22] Eero P Simoncelli and Bruno A Olshausen. Natural image statistics and neural representation. *Annual Review of Neuroscience*, 24(1):1193–1216, 2001. PMID: 11520932.

[23] Marius Pachitariu, Carsen Stringer, and Kenneth D Harris. Robustness of spike deconvolution for neuronal calcium imaging. *Journal of Neuroscience*, 38(37):7976–7985, 2018.

[24] B. Bordelon, A. Canatar, and C. Pehlevan. Spectrum dependent learning curves in kernel regression and wide neural networks. *International Conference of Machine Learning*, 2020.

[25] Abdulkadir Canatar, Blake Bordelon, and Cengiz Pehlevan. Spectral bias and task-model alignment explain generalization in kernel regression and infinitely wide neural networks, 2020.

[26] Mark Stopfer, Vivek Jayaraman, and Gilles Laurent. Intensity versus identity coding in an olfactory system. *Neuron*, 39(6):991–1004, 2003.

[27] Saul Kato, Harris S Kaplan, Tina Schrödel, Susanne Skora, Theodore H Lindsay, Eviatar Yemini, Shawn Lockery, and Manuel Zimmer. Global brain dynamics embed the motor command sequence of caenorhabditis elegans. *Cell*, 163(3):656–669, 2015.

[28] Brice Bathellier, Derek L. Buhl, Riccardo Accolla, and Alan Carleton. Dynamic ensemble odor coding in the mammalian olfactory bulb: Sensory information at different timescales. *Neuron*, 57(4):586 – 598, 2008.

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

[30] Juan A Gallego, Matthew G Perich, Stephanie N Naufel, Christian Ethier, Sara A Solla, and Lee E Miller. Cortical population activity within a preserved neural manifold underlies multiple motor behaviors. *Nature communications*, 9(1):1–13, 2018.

[31] T. Patrick Sadtler, M. Kristen Quick, D. Matthew Golub, M. Steven Chase, I. Steven Ryu, C. Elizabeth Tyler-Kabara, M. Byron Yu, and P. Aaron Batista. Neural constraints on learning. *Nature*, 512(5500):423–426, 2014.

[32] Ashok Litwin-Kumar, Kameron Decker Harris, Richard Axel, Haim Sompolinsky, and L.F. Abbott. Optimal degrees of synaptic connectivity. *Neuron*, 2017.

[33] Peiran Gao, Eric Trautmann, Byron Yu, Gopal Santhanam, Stephen Ryu, Krishna Shenoy, and Surya Ganguli. A theory of multineuronal dimensionality, dynamics and measurement. *bioRxiv*, 2017.

[34] Peiran Gao and Surya Ganguli. On simplicity and complexity in the brave new world of large-scale neuroscience. *Current Opinion in Neurobiology*, 32:148 – 155, 2015. Large-Scale Recording Technology (32).

[35] James Mercer. Xvi. functions of positive and negative type, and their connection the theory of integral equations. *Philosophical transactions of the royal society of London. Series A, containing papers of a mathematical or physical character*, 209(441-458):415–446, 1909.

[36] D Hansel and C Van Vreeswijk. How noise contributes to contrast invariance of orientation tuning in cat visual cortex. *Journal of Neuroscience*, 22(12):5118–5128, 2002.

[37] Kenneth D Miller and Todd W Troyer. Neural noise can explain expansive, power-law nonlinearities in neural response functions. *Journal of neurophysiology*, 87(2):653–659, 2002.

[38] Nicholas J Priebe, Ferenc Mechler, Matteo Carandini, and David Ferster. The contribution of spike threshold to the dichotomy of cortical simple and complex cells. *Nature neuroscience*, 7(10):1113–1122, 2004.

[39] Nicholas J Priebe and David Ferster. Inhibition, spike threshold, and stimulus selectivity in primary visual cortex. *Neuron*, 57(4):482–497, 2008.

[40] E. Adelson and J. Bergen. Spatiotemporal energy models for the perception of motion. *Journal of the Optical Society of America. A, Optics and image science*, 2 2:284–99, 1985.

[41] Eero P. Simoncelli and David J. Heeger. A model of neuronal responses in visual area mt. *Vision Research*, 38(5):743–761, 1998.

[42] Jonathan Dong, Ruben Ohana, Mushegh Rafayelyan, and Florent Krzakala. Reservoir computing meets recurrent kernels and structured transforms, 2020.

[43] Greg Yang. Tensor programs i: Wide feedforward or recurrent neural networks of any architecture are gaussian processes, 2019.

[44] Greg Yang. Tensor programs ii: Neural tangent kernel for any architecture, 2020.

[45] Joaquin M. Fuster and Garrett E. Alexander. Neuron activity related to short-term memory. *Science*, 173(3997):652–654, 1971.

[46] Earl K. Miller, Cynthia A. Erickson, and Robert Desimone. Neural mechanisms of visual working memory in prefrontal cortex of the macaque. *Journal of Neuroscience*, 16(16):5154–5167, 1996.

[47] H. Sompolinsky, A. Crisanti, and H. J. Sommers. Chaos in random neural networks. *Phys. Rev. Lett.*, 61:259–262, Jul 1988.

[48] Matthew Botvinick, Sam Ritter, Jane Wang, Zeb Kurth-Nelson, Charles Blundell, and Demis Hassabis. Optimal degrees of synaptic connectivity. *Trends in Cognitive Sciences*, 23:408–422, 2019.

[49] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate, 2016.

[50] Yann LeCun and Yoshua Bengio. *Convolutional Networks for Images, Speech, and Time Series*, page 255–258. MIT Press, Cambridge, MA, USA, 1998.

[51] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 25, pages 1097–1105. Curran Associates, Inc., 2012.

[52] Taco S. Cohen and Max Welling. Group equivariant convolutional networks, 2016.

[53] Florian Meier, Raphaël Dang-Nhu, and Angelika Steger. Adaptive tuning curve widths improve sample efficient learning. *Frontiers in Computational Neuroscience*, 14, 2020.

[54] Mandyam Veerambudi Srinivasan, Simon Barry Laughlin, and Andreas Dubs. Predictive coding: a fresh view of inhibition in the retina. *Proceedings of the Royal Society of London. Series B. Biological Sciences*, 216(1205):427–459, 1982.

[55] Johannes H van Hateren. A theory of maximizing sensory information. *Biological cybernetics*, 68(1):23–29, 1992.

[56] Rajesh PN Rao and Dana H Ballard. Predictive coding in the visual cortex: a functional interpretation of some extra-classical receptive-field effects. *Nature neuroscience*, 2(1):79–87, 1999.

[57] Matthew Chalk, Olivier Marre, and Gasper Tkacik. Toward a unified theory of efficient, predictive, and sparse coding. *Proceedings of the National Academy of Sciences*, 115(1):186–191, 2018.

[58] S. Edelman. Representation is representation of similarities. *The Behavioral and brain sciences*, 21 4:449–67; discussion 467–98, 1998.

[59] Nikolaus Kriegeskorte, Marieke Mur, and Peter Bandettini. Representational similarity analysis - connecting the branches of systems neuroscience. *Frontiers in Systems Neuroscience*, 2:4, 2008.

[60] Aarre Laakso. Content and cluster analysis: Assessing representational similarity in neural systems. *Philosophical Psychology*, 13, 05 2000.

[61] Simon Kornblith, Mohammad Norouzi, Honglak Lee, and Geoffrey Hinton. Similarity of neural network representations revisited, 2019.

[62] Charles F. Cadieu, Ha Hong, Daniel L. K. Yamins, Nicolas Pinto, Diego Ardila, Ethan A. Solomon, Najib J. Majaj, and James J. DiCarlo. Deep neural networks rival the representation of primate it cortex for core visual object recognition. *PLOS Computational Biology*, 10(12):1–18, 12 2014.

[63] Cengiz Pehlevan, Anirvan M. Sengupta, and Dmitri B. Chklovskii. Why do similarity matching objectives lead to hebbian/anti-hebbian networks? *Neural Computation*, 30(1):84–124, 2018. PMID: 28957017.

[64] Jeremy E. Niven and Simon B. Laughlin. Energy limitation as a selective pressure on the evolution of sensory systems. *Journal of Experimental Biology*, 211(11):1792–1804, 2008.

[65] Tomas Hromadka, Michael R DeWeese, and Anthony M Zador. Sparse representation of sounds in the unanesthetized auditory cortex. *PLOS Biology*, 6(1):1–14, 01 2008.

[66] John K Chapin and Miguel A.L Nicolelis. Principal component analysis of neuronal ensemble activity reveals multidimensional somatosensory representations. *Journal of Neuroscience Methods*, 94(1):121 – 140, 1999.

[67] Xaq Pitkow and Markus Meister. Decorrelation and efficient coding by retinal ganglion cells. *Nature neuroscience*, 15(4):628–635, 2012.

[68] Reza Abbasi-Asl, Cengiz Pehlevan, Bin Yu, and Dmitri Chklovskii. Do retinal ganglion cells project natural scenes to their principal subspace and whiten them? In *2016 50th Asilomar Conference on Signals, Systems and Computers*, pages 1641–1645. IEEE, 2016.

[69] J. Gallego, M. Perich, S. Naufel, C. Ethier, S. Solla, and L. Miller. Cortical population activity within a preserved neural manifold underlies multiple motor behaviors. *Nature Communications*, 9, 10 2018.

[70] M Haft and J Leo van Hemmen. Theory and implementation of infomax filters for the retina. *Network: Computation in Neural Systems*, 9(1):39–71, 1998.

[71] Yanping Huang and Rajesh PN Rao. Predictive coding. *Wiley Interdisciplinary Reviews: Cognitive Science*, 2(5):580–593, 2011.

[72] Bruno Averbeck, Peter Latham, and Alexandre Pouget. Neural correlations, population coding and computation. *Nature Reviews Neuroscience*, 7, 2006.

[73] Marlene R Cohen and Adam Kohn. Measuring and interpreting neuronal correlations. *Nature neuroscience*, 14(7):811, 2011.

[74] Peter L. Bartlett, Philip M. Long, Gábor Lugosi, and Alexander Tsigler. Benign overfitting in linear regression. *Proceedings of the National Academy of Sciences*, 117(48):30063–30070, 2020.

[75] Trevor Hastie, Andrea Montanari, Saharon Rosset, and Ryan J. Tibshirani. Surprises in high-dimensional ridgeless least squares interpolation, 2020.

[76] Mikhail Belkin, Daniel Hsu, Siyuan Ma, and Soumik Mandal. Reconciling modern machine-learning practice and the classical bias–variance trade-off. *Proceedings of the National Academy of Sciences*, 116(32):15849–15854, 2019.

[77] Bernard Widrow and Marcian E. Hoff. Adaptive switching circuits. In *1960 IRE WESCON Convention Record, Part 4*, pages 96–104, New York, 1960. IRE.

[78] John Hertz, Anders Krough, and Richard Palmer. *Introduction To The Theory Of Neural Computation*, volume 44. 01 1991.

[79] Peter Sollich. Learning curves for gaussian processes. In *Neurips*, 1998.

[80] 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, pages 519–526. MIT Press, 2002.

[81] 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, 1997.

[82] Oleg Rumyantsev, Jérôme Lecoq, Oscar Hernandez, Yanping Zhang, Joan Savall, Radosław Chrapkiewicz, Jane Li, Hongkui Zeng, Surya Ganguli, and Mark Schnitzer. Fundamental bounds on the fidelity of sensory cortical coding. *Nature*, 580:1–6, 04 2020.

[83] James Bradbury, Roy Frostig, Peter Hawkins, Matthew James Johnson, Chris Leary, Dougal Maclaurin, George Necula, Adam Paszke, Jake VanderPlas, Skye Wanderman-Milne, and Qiao Zhang. JAX: composable transformations of Python+NumPy programs, 2018.

Supplementary Information

Function Space Decomposition of Population Responses

SVD of population responses is usually evaluated with respect to a discrete and finite set of stimuli. In the main paper, we implicitly assumed that a generalization of SVD to a continuum of stimuli. In this section we justify this assumption using techniques from functional analysis. This also justifies our use of SVD in Figure 4. Mercer's theorem guarantees the existence of an eigendecomposition of any inner product kernel $K(\boldsymbol{\theta}, \boldsymbol{\theta}')$ in terms of a complete orthonormal set of functions $\{\phi_k\}_{k=1}^{\infty}$ [17]. In particular, there exist a non-negative (but possibly zero) summable eigenvalues $\{\lambda_k\}_{k=1}^{\infty}$ and a corresponding set of orthonormal eigenfunctions such that

$$K(\boldsymbol{\theta}, \boldsymbol{\theta}') = \sum_{k=1}^{\infty} \lambda_k \phi_k(\boldsymbol{\theta}) \phi_k(\boldsymbol{\theta}'). \tag{SI.1}$$

For a stimulus distribution $p(\boldsymbol{\theta})$, the set of functions $\{\phi_k\}_{k=1}^{\infty}$ are orthonormal and form a complete basis for square integrable functions L_2 which means

$$\begin{aligned} \langle \phi_k(\boldsymbol{\theta}) \phi_\ell(\boldsymbol{\theta}) \rangle_{\boldsymbol{\theta}} &= \int p(\boldsymbol{\theta}) \phi_k(\boldsymbol{\theta}) \phi_\ell(\boldsymbol{\theta}) d\boldsymbol{\theta} = \delta_{k\ell} \\ f(\boldsymbol{\theta}) &= \sum_k \left\langle f(\boldsymbol{\theta}') \phi(\boldsymbol{\theta}') \right\rangle_{\boldsymbol{\theta}'} \phi_k(\boldsymbol{\theta}) \ , \ \forall f \in L_2. \end{aligned} \tag{SI.2}$$

We will now identify the necessary conditions for these functions to be the basis of orthonormal kernel eigenfunctions for the population code, which will allow us to express the SVD of the population code. Each of the tuning curves r_i can be expressed in this basis with $K \leq N$ of the functions in the set $\{\phi_k\}_{k=1}^{\infty}$

$$r_i(\boldsymbol{\theta}) = \sum_{k=1}^{K} A_{ik} \phi_k(\boldsymbol{\theta}), \tag{SI.3}$$

where we introduced a matrix $\mathbf{A} \in \mathbb{R}^{N \times K}$ of expansion coefficients. Note that $\text{rank}(\mathbf{A}) \leq \min\{N, K\}$. We compute the singular value decomposition of the finite matrix $\mathbf{A}$

$$\mathbf{A} = \sqrt{N} \sum_{k=1}^{\text{rank}(\mathbf{A})} \sqrt{\lambda_k} \mathbf{u}_k \mathbf{v}_k^\top. \tag{SI.4}$$

The signal correlation matrix for this population code can be computed in closed form

$$\boldsymbol{\Sigma}_s = \frac{1}{N} \mathbf{A} \left\langle \boldsymbol{\phi}(\boldsymbol{\theta}) \boldsymbol{\phi}(\boldsymbol{\theta})^\top \right\rangle_{\boldsymbol{\theta}} \mathbf{A}^\top = \frac{1}{N} \mathbf{A} \mathbf{A}^\top = \sum_{k=1}^{\text{rank}(\mathbf{A})} \lambda_k \mathbf{u}_k \mathbf{u}_k^\top, \tag{SI.5}$$

due to the orthonormality of $\{\phi_k\}$. Thus the principal axes $\mathbf{u}_k$ of the neural correlations are the left singular vectors of $\mathbf{A}$. We may similarly express the inner product kernel in terms of the eigenfunctions

$$K(\boldsymbol{\theta}, \boldsymbol{\theta}') = \frac{1}{N} \boldsymbol{r}(\boldsymbol{\theta}) \cdot \boldsymbol{r}(\boldsymbol{\theta}') = \frac{1}{N} \boldsymbol{\phi}(\boldsymbol{\theta})^\top \mathbf{A}^\top \mathbf{A} \boldsymbol{\phi}(\boldsymbol{\theta}'). \tag{SI.6}$$

The kernel eigenvalue problem demands [17]

$$\int p(\boldsymbol{\theta}) K(\boldsymbol{\theta}, \boldsymbol{\theta}') \boldsymbol{\phi}(\boldsymbol{\theta}) d\boldsymbol{\theta} = \frac{1}{N} \mathbf{A}^\top \mathbf{A} \boldsymbol{\phi}(\boldsymbol{\theta}') = \boldsymbol{\Lambda} \boldsymbol{\phi}(\boldsymbol{\theta}') \implies \frac{1}{N} \mathbf{A}^\top \mathbf{A} = \boldsymbol{\Lambda} \tag{SI.7}$$

$$\implies \sum_{k=1}^{\text{rank}(\mathbf{A})} \lambda_k \mathbf{v}_k \mathbf{v}_k^\top = \sum_{k=1}^{\text{rank}(\mathbf{A})} \lambda_k \boldsymbol{e}_k \boldsymbol{e}_k^\top. \tag{SI.8}$$

For $\{\phi_k\}$ to be the set of kernel eigenfunctions, the $\mathbf{v}_k$ vectors must be identical to $\boldsymbol{e}_k$, the Cartesian unit vectors. From this exercise, we find that the SVD for $\mathbf{A}$ has the form $\mathbf{A} = \sqrt{N}\sum_{k=1}^{\text{rank}(\mathbf{A})}\sqrt{\lambda_k}\mathbf{u}_k\boldsymbol{e}_k^\top$ With this choice, the population code admits a singular value decomposition

$$\boldsymbol{r}(\boldsymbol{\theta}) = \mathbf{A}\boldsymbol{\phi}(\boldsymbol{\theta}) = \sqrt{N}\sum_{k=1}^{\text{rank}(\mathbf{A})}\sqrt{\lambda_k}\mathbf{u}_k\phi_k(\boldsymbol{\theta}). \tag{SI.9}$$

This singular value decomposition demonstrates the connection between neural manifold structure (principal axes $\mathbf{u}_k$) and function approximation (kernel eigenfunctions ϕ_k). This singular value decomposition can be verified by computing the inner product kernel and the correlation matrix, utilizing the orthonormality of $\{\mathbf{u}_k\}$ and $\{\phi_k\}$. For a finite number of neurons N the space of learnable functions is at most N dimensional since linear readouts lie in $\text{span}\{r_i(\boldsymbol{\theta})\}_{i=1}^N$, an $K \leq N$ dimensional subspace of L_2. This is the reason why $\text{rank}(\mathbf{A}) \leq N$ and why the Mercer decomposition of the kernel can be truncated after N nonzero eigenvalues.

Finite Stimulus Spaces: Finding Eigenfunctions with SVD

The singular value decomposition provides a link between the dimensionality of the population code (correlation structure, $\mathbf{u}_k$) and the functional properties of downstream neurons (kernel structure ϕ_k). The equivalence also holds for recordings with finite stimuli and finite neurons as can be seen by SVD of the matrix $\mathbf{R} = \mathbf{U}\boldsymbol{\Lambda}^{1/2}\boldsymbol{\Phi}^\top$ with entries $R_{i\mu} = r_i(\boldsymbol{\theta}^\mu)$. To formalize this notion, we can introduce a Monte-Carlo estimate of the integral eigenvalue problem by introducing an empirical measure on P samples $\hat{p}(\boldsymbol{\theta}) = \frac{1}{P}\sum_{\mu=1}^P \delta(\boldsymbol{\theta} - \boldsymbol{\theta}^\mu)$. With this choice

$$\left\langle K(\boldsymbol{\theta},\boldsymbol{\theta}')\phi_k(\boldsymbol{\theta})\right\rangle_{\boldsymbol{\theta}\sim p(\boldsymbol{\theta})} = \int \hat{p}(\boldsymbol{\theta})K(\boldsymbol{\theta},\boldsymbol{\theta}')\phi_k(\boldsymbol{\theta})d\boldsymbol{\theta} = \frac{1}{P}\sum_{\mu=1}^P K(\boldsymbol{\theta}^\mu,\boldsymbol{\theta}')\phi_k(\boldsymbol{\theta}^\mu) = \lambda_k\phi_k(\boldsymbol{\theta}'). \tag{SI.10}$$

Demanding this equality for $\boldsymbol{\theta}' = \boldsymbol{\theta}^\nu$ $\nu = 1, ..., P$ generates a matrix eigenvalue problem

$$\mathbf{K}\boldsymbol{\Phi} = P\boldsymbol{\Phi}\boldsymbol{\Lambda}. \tag{SI.11}$$

The eigenfunctions restricted to $\{\boldsymbol{\theta}^\mu\}$ are identified as the columns of $\boldsymbol{\Phi}$ while the eigenvalues are the diagonal elements of $\boldsymbol{\Lambda}_{k\ell} = \lambda_k\delta_{k\ell}$. For the case where N and P are finite, the spectrum obtained through eigendecomposition of the kernel $\mathbf{K}$ is the same as would be obtained through the finite N signal correlation matrix $\boldsymbol{\Sigma}_s$, since they are inner and outer products of trial averaged population response matrices $\boldsymbol{R}$.

Generalization in Kernel Regression

Recent work has established analytic results that predict the average case generalization error for kernel regression

$$E_g = \langle E_g(\mathcal{D})\rangle_{\mathcal{D}} = \left\langle (f(\boldsymbol{\theta},\mathcal{D}) - y(\boldsymbol{\theta}))^2\right\rangle_{\boldsymbol{\theta},\mathcal{D}} \tag{SI.12}$$

where $E_g(\mathcal{D}) = \left\langle (f(\boldsymbol{\theta},\mathcal{D}) - y(\boldsymbol{\theta}))^2\right\rangle_{\boldsymbol{\theta}}$ is the generalization error for a certain sample $\mathcal{D}$ of size P and $f(\boldsymbol{\theta},\mathcal{D})$ is the kernel regression solution for $\mathcal{D}$ [24, 25]. The typical or average case error E_g is obtained by averaging over all possible datasets of size P. This average case generalization error is determined solely by the decomposition of the target function $y(\mathbf{x})$ along the eigenbasis of the kernel and the eigenspectrum of the kernel. This diagonalization takes the form

$$\int p(\boldsymbol{\theta})K(\boldsymbol{\theta},\boldsymbol{\theta}')\phi_k(\boldsymbol{\theta})d\boldsymbol{\theta} = \lambda_k\phi_k(\boldsymbol{\theta}') \tag{SI.13}$$

Since the eigenfunctions form a complete set of square integrable functions, we expand both the target function $y(\boldsymbol{\theta})$ and the learned function $f(\boldsymbol{\theta})$ in this basis

$$y(\boldsymbol{\theta}) = \sum_k v_k \phi_k(\boldsymbol{\theta}) \ , \ f(\boldsymbol{\theta}) = \sum_k w_k \phi_k(\boldsymbol{\theta}) \tag{SI.14}$$

Due to the orthonormality of the kernel eigenfunctions $\{\phi_k\}$, the generalization error for any set of coefficients $\mathbf{w}$ is

$$E_g(\mathbf{w}) = \left\langle (y(\boldsymbol{\theta}) - f(\boldsymbol{\theta}))^2 \right\rangle_{\boldsymbol{\theta}} = \sum_k (w_k - v_k)^2 = ||\mathbf{w} - \mathbf{v}||^2 \tag{SI.15}$$

We now introduce training error, or empirical loss, which depends on the disorder in the dataset $\mathcal{D} = \{(\boldsymbol{\theta}^\mu, y^\mu)\}_{\mu=1}^P$

$$H(\mathbf{w}, \mathcal{D}) = \sum_\mu (\mathbf{w} \cdot \boldsymbol{\phi}(\boldsymbol{\theta}^\mu) - \mathbf{v} \cdot \boldsymbol{\phi}(\boldsymbol{\theta}^\mu))^2 + \lambda \sum_k \frac{w_k^2}{\lambda_k} \tag{SI.16}$$

It is straightforward to verify that the optimal $\mathbf{w}^*$ which minimizes $H(\mathbf{w}, \mathcal{D})$ is the kernel regression solution for kernel with eigenvalues $\{\lambda_k\}$ when $\lambda \to 0$. The optimal weights $\mathbf{w}$ can be identified through the first order condition $\nabla H(\mathbf{w}, \mathcal{D}) = 0$ which gives

$$\mathbf{w}^* = (\boldsymbol{\Phi}\boldsymbol{\Phi}^\top + \lambda \boldsymbol{\Lambda}^{-1})^{-1} \boldsymbol{\Phi}\boldsymbol{\Phi}^\top \mathbf{v} = \mathbf{v} - \lambda (\boldsymbol{\Phi}\boldsymbol{\Phi}^\top + \lambda \boldsymbol{\Lambda}^{-1})^{-1} \boldsymbol{\Lambda}^{-1} \mathbf{v} \tag{SI.17}$$

where $\Phi_{k,\mu} = \phi_k(\mathbf{x}^\mu)$ are the eigenfunctions evaluated on the training data and $\Lambda_{k,\ell} = \delta_{k,\ell}\lambda_k$ is a a diagonal matrix containing the kernel eigenvalues. The generalization error for this optimal solution is

$$E_g(\mathcal{D}) = ||\mathbf{w}^* - \mathbf{v}||^2 = \mathbf{v}^\top \boldsymbol{\Lambda}^{-1} \mathbf{G}(\mathcal{D})^2 \boldsymbol{\Lambda}^{-1} \mathbf{v} \ , \ \mathbf{G}(\mathcal{D}) = \left(\frac{1}{\lambda} \boldsymbol{\Phi}\boldsymbol{\Phi}^\top + \boldsymbol{\Lambda}^{-1} \right)^{-1} \tag{SI.18}$$

We note that the dependence on the randomly sampled dataset $\mathcal{D}$ only appears through the matrix $\mathbf{G}(\mathcal{D})$. Thus to compute the *typical* generalization error we need to average over this matrix $\langle \mathbf{G}(\mathcal{D}) \rangle_{\mathcal{D}}$. There are multiple strategies to perform such an average and we will study one here based on a partial differential equation which was introduced in [79, 80] and studied further in [24, 25]. In this setting, we denote the average matrix $\mathbf{G}(P) = \langle \mathbf{G}(\mathcal{D}) \rangle_{|\mathcal{D}|=P}$ for a dataset of size P. We first will derive a recursion relationship using the Sherman Morrison formula for a rank-1 update to an inverse matrix. We imagine adding a new sampled feature vector ϕ to a dataset $\boldsymbol{\Phi}$ with size P. The average matrix $\mathbf{G}(P+1)$ at $P+1$ samples can be related to $\mathbf{G}(P)$ through the Sherman Morrison rule

$$\mathbf{G}(P+1) = \left\langle \left(\frac{1}{\lambda} \boldsymbol{\Phi}\boldsymbol{\Phi}^\top + \frac{1}{\lambda} \phi\phi^\top + \boldsymbol{\Lambda}^{-1} \right)^{-1} \right\rangle_{\phi,\mathcal{D}} = \mathbf{G}(P) - \left\langle \frac{\mathbf{G}(\mathcal{D})\phi\phi^\top \mathbf{G}(\mathcal{D})}{\lambda + \phi^\top \mathbf{G}(\mathcal{D})\phi} \right\rangle_{\phi,\mathcal{D}} \tag{SI.19}$$

$$\approx \mathbf{G}(P) - \frac{\left\langle \mathbf{G}(\mathcal{D}) \left\langle \phi\phi^\top \right\rangle_\phi \mathbf{G}(\mathcal{D}) \right\rangle_{\mathcal{D}}}{\lambda + \langle \phi^\top \mathbf{G}(\mathcal{D})\phi \rangle_{\phi,\mathcal{D}}} \tag{SI.20}$$

where in the last step we approximated the average of the ratio with the ratio of averages. This operation, is of course, unjustified theoretically, but has been shown to produce accurate learning curves [24, 80]. Since the chosen basis of kernel eigenfunctions are orthonormal, the average over the new sample is trivial $\left\langle \phi\phi^\top \right\rangle_\phi = \mathbf{I}$. We thus arrive at the following recursion relationship for $\mathbf{G}$

$$\mathbf{G}(P+1) = \mathbf{G}(P) - \frac{\left\langle \mathbf{G}(\mathcal{D})^2 \right\rangle_{\mathcal{D}}}{\lambda + \mathrm{Tr}\ \mathbf{G}(P)} \tag{SI.21}$$

By introducing an additional source J so that $\mathbf{G}(\mathcal{D},J)^{-1} = \frac{1}{\lambda}\mathbf{\Phi}\mathbf{\Phi}^\top + \mathbf{\Lambda}^{-1} + J\mathbf{I}$, we can relate $\mathbf{G}(\mathcal{D},J)$'s first and second moments through differentiation

$$\frac{\partial}{\partial J}\mathbf{G}(P,J) = \frac{\partial}{\partial J}\left\langle \left(\frac{1}{\lambda}\mathbf{\Phi}\mathbf{\Phi}^\top + J\mathbf{I} + \mathbf{\Lambda}^{-1}\right)^{-1}\right\rangle_{\mathcal{D}} = -\left\langle \mathbf{G}(\mathcal{D},J)^2\right\rangle_{\mathcal{D}} \tag{SI.22}$$

Thus the recursion relation simplifies to

$$\mathbf{G}(P+1,J) - \mathbf{G}(P,J) \approx \frac{\partial}{\partial p}\mathbf{G}(p,J) = \frac{1}{\lambda + \mathrm{Tr}\mathbf{G}(P,J)}\frac{\partial}{\partial J}\mathbf{G}(P,J) \tag{SI.23}$$

where we approximated the finite difference in P as a derivative, treating P as a continuous variable. Taking the trace of both sides and defining $\kappa(P,J) = \lambda + \mathrm{Tr}\mathbf{G}(P,J)$ we arrive at the following quasilinear PDE

$$\frac{\partial}{\partial P}\kappa(P,J) = \frac{1}{\kappa(P,J)}\frac{\partial}{\partial J}\kappa(P,J) \tag{SI.24}$$

with the initial condition $\kappa(0,J) = \lambda + \mathrm{Tr}(\mathbf{\Lambda}^{-1} + J\mathbf{I})^{-1}$. Using the method of characteristics, we arrive at the solution $\kappa(P,J) = \lambda + \mathrm{Tr}\left(\mathbf{\Lambda}^{-1} + (v + \frac{P}{\kappa(P,J)})\mathbf{I}\right)^{-1}$. Using this solution to κ, we can identify the solution to $\mathbf{G}$

$$\mathbf{G}(P,J)_{k,\ell} = \left(\frac{P}{\kappa} + J + \lambda_k^{-1}\right)^{-1}\delta_{k,\ell} = \frac{\kappa\lambda_k}{\lambda_k P + \kappa + J\kappa\lambda_k}\delta_{k,\ell} \tag{SI.25}$$

The generalization error, therefore can be written as

$$E_g = \mathbf{v}^\top\mathbf{\Lambda}^{-1}\left\langle \mathbf{G}(\mathcal{D})^2\right\rangle_{\mathcal{D}}\mathbf{\Lambda}^{-1}\mathbf{v} = -\frac{\partial}{\partial J}\mathbf{v}^\top\mathbf{\Lambda}^{-1}\mathbf{G}(P,J)\mathbf{\Lambda}^{-1}\mathbf{v} \tag{SI.26}$$

$$= -\sum_k \frac{v_k^2}{\lambda_k^2}\frac{\partial}{\partial J}\left(\frac{P}{\kappa} + J + \lambda_k^{-1}\right)^{-1} = \frac{\kappa^2}{1-\gamma}\sum_k \frac{v_k^2}{(\lambda_k P + \kappa)^2} \tag{SI.27}$$

where $\gamma = P\sum_k \frac{\lambda_k^2}{(\lambda_k P+\kappa)^2}$, giving the desired result. Note that κ depends on J implicitly, which is the source of the $\frac{1}{1-\gamma}$ factor. This result was recently reproduced using techniques from statistical mechanics [24, 25].

Translation Invariant Kernels

For the special case where the data distribution $p(\boldsymbol{\theta}) = \frac{1}{V}$ is uniform over volume V and the kernel is translation invariant $K(\boldsymbol{\theta},\boldsymbol{\theta}') = \kappa(\boldsymbol{\theta}-\boldsymbol{\theta}')$, the kernel can be diagonalized in the basis of plane waves

$$\int p(\boldsymbol{\theta})K(\boldsymbol{\theta},\boldsymbol{\theta}')\phi_{\mathbf{k}}(\boldsymbol{\theta})d\boldsymbol{\theta} = \frac{1}{V}\int \kappa(\boldsymbol{\theta}-\boldsymbol{\theta}')e^{i\mathbf{k}\cdot\boldsymbol{\theta}}d\boldsymbol{\theta} = \frac{1}{V}\hat{\kappa}(\mathbf{k})e^{i\mathbf{k}\cdot\boldsymbol{\theta}'} \tag{SI.28}$$

The eigenvalues are the Fourier components of the Kernel $\lambda_{\mathbf{k}} = \frac{1}{V}\hat{\kappa}(\mathbf{k}) = \frac{1}{V}\int d\boldsymbol{\theta}e^{i\mathbf{k}\cdot\boldsymbol{\theta}}\kappa(\boldsymbol{\theta})$ while the eigenfunctions are plane waves $\phi_{\mathbf{k}}(\boldsymbol{\theta}) = e^{i\mathbf{k}\cdot\boldsymbol{\theta}}$. The set of admissible momenta $\mathcal{S}_{\mathbf{k}} = \{\mathbf{k}_0, \pm\mathbf{k}_1, \pm\mathbf{k}_2, ...\}$ are determined by the boundary conditions. The diagonalized representation of the kernel is therefore

$$K(\boldsymbol{\theta},\boldsymbol{\theta}') = \sum_{\mathbf{k}\in\mathcal{S}_{\mathbf{k}}}\lambda_{\mathbf{k}}e^{i\mathbf{k}\cdot(\boldsymbol{\theta}-\boldsymbol{\theta}')} \tag{SI.29}$$

For example, if the space is the torus $\mathbb{T}^n = S^1 \times S^1 \times ... \times S^1$, then the space of admissable momenta are the points on the integer lattice $\mathcal{S}_{\mathbf{k}} = \mathbb{Z}^n = \{\mathbf{k}\in\mathbb{R}^n | k_i \in \mathbb{Z}\ \forall i = 1, ..., n\}$. Reality and

symmetry of the kernel demand that $\mathrm{Im}(\lambda_{\mathbf{k}}) = 0$ and $\lambda_{-\mathbf{k}} = \lambda_{\mathbf{k}} \geq 0$. Most of the models in this paper consider $\theta \sim \mathrm{Unif}\left(S^1\right)$, where the kernel has the following Fourier/Mercer decomposition

$$\begin{aligned} K(\theta - \theta') &= \sum_{k=-\infty}^{\infty} \lambda_k e^{ik(\theta-\theta')} = 2\sum_{k=0}^{\infty} \lambda_k \cos(k(\theta - \theta')) \\ &= \sum_{k=0}^{\infty} \lambda_k \left[\sqrt{2}\cos(k\theta)\sqrt{2}\cos(k\theta') + \sqrt{2}\sin(k\theta)\sqrt{2}\sin(k\theta')\right] \end{aligned} \tag{SI.30}$$

where we invoked the simple trigonometric identity $\cos(a-b) = \cos(a)\cos(b) + \sin(a)\sin(b)$. By recognizing that $\{\sqrt{2}\cos(k\theta), \sqrt{2}\sin(k\theta)\}_{k=0}^{\infty}$ form a complete orthonormal set of functions with respect to $\mathrm{Unif}\left(S^1\right)$, we have identified this as the collection of kernel eigenfunctions.

Visualization of Feedforward Gabor V1 Model and Induced Kernels

Examples of the induced kernels for the Gabor-bank V1 model are provided in Figure SI.2. We show how choice of rectifying nonlinearity $\psi(z)$ and sparsifying threshold a influence the kernel and their spectra. Learning curves for simple orientation tasks are provided.

Supplementary Figures

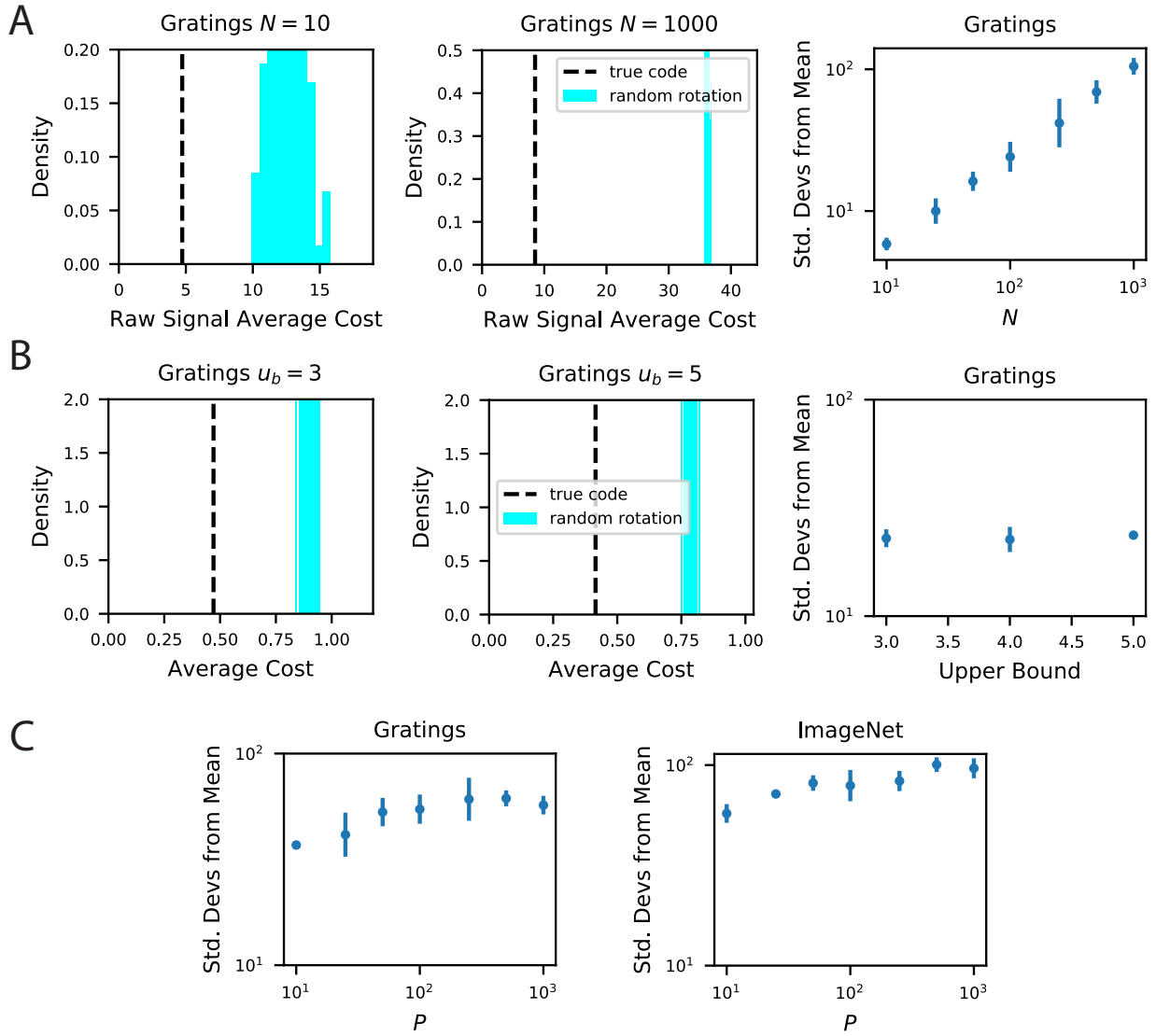

Figure SI.1: Our metabolic efficiency finding is robust to different pre-processing techniques and upper bounds on neural firing. **A** We show the same result as in Figure 3 except we use raw (non z-scored) estimate of responses for each stimulus. **B** Our result is robust to imposition of firing rate upper bounds u_b on each neuron. This result uses the z-scored responses to be consistent with the rest of the paper. The biological code achieves a maximum z-score values in the range $[3.2, 4.7]$, which motivated the range of our tested upper bound values $\{3, 4, 5\}$. **C** Our finding is robust to the number of sampled stimuli P as we show in an experiment where rotations in $N = 500$ dimensional subspace.

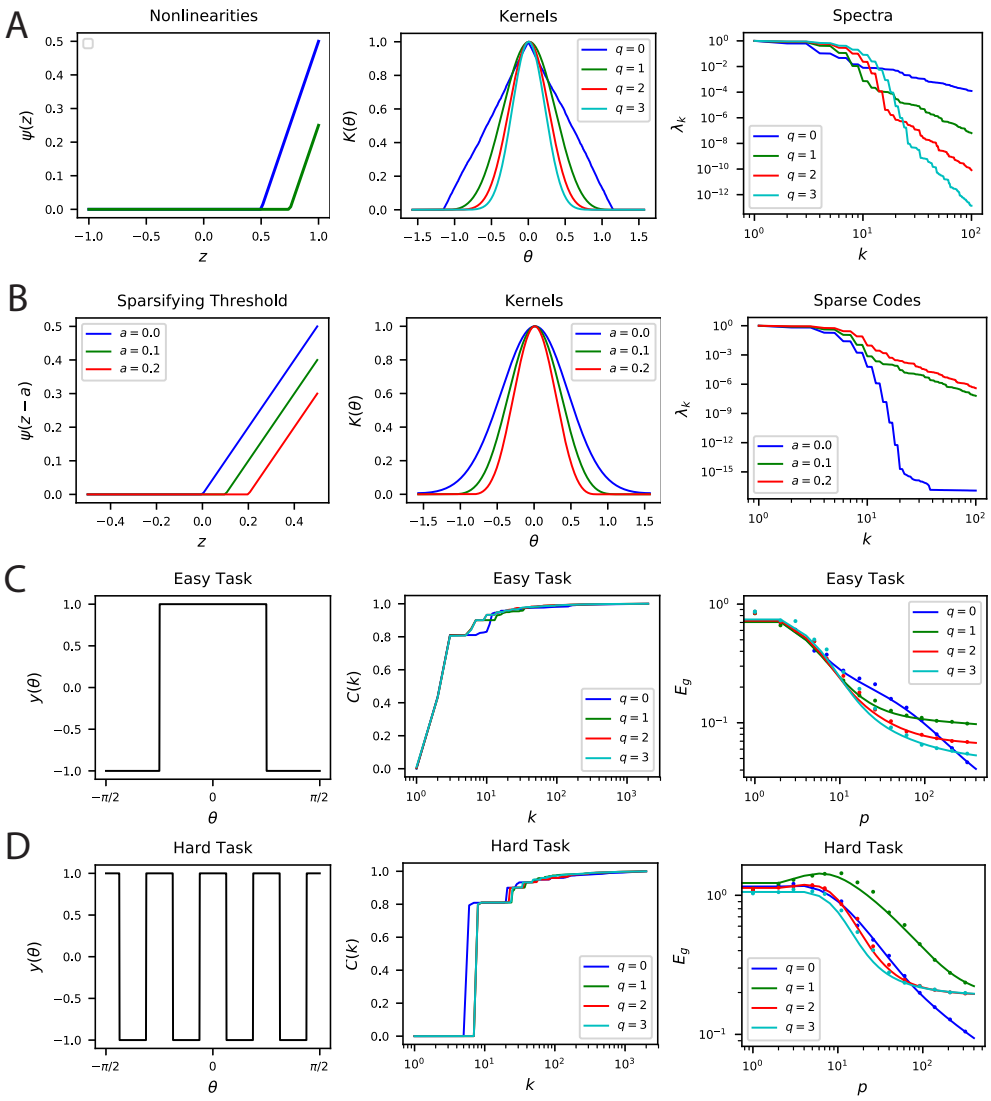

Figure SI.2: Nonlinear Rectification and proportion of simple and complex cells influences the inductive bias of the population code. **A** The choice of nonlinearity has influence on the kernel and its spectrum. If the nonlinearity is $\psi(z) = \max(0, z^q)$, then $\lambda_k \sim k^{-2n-2}$. **B** The sparsity can be increased by shifting the nonlinearity $\psi(z) \to \psi(z-a)$. Sparser codes have higher dimensionality. **C-D** Easy and hard orientation discrimination tasks with varying nonlinear polynomial order q. At low sample sizes, large q performs better, whereas at large P, the step function nonlinearity $q = 0$ achieves the best performance.