YOU DON'T NEED FEATURE LEARNING TO DO IN-CONTEXT LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

How should we understand memorization vs. generalization in in-context learning (ICL) tasks, and when does one happen vs. the other? To investigate, we study how rotation-invariant kernel machines learn the ICL linear regression task proposed by Raventós et al. (2023). By decomposing the task in the kernel eigenbasis, we obtain a spectral picture of learning in this ICL setting. This picture reveals that as sample size grows, kernel machines learn successively-higher-degree components of the memorizing solution, which depend upon successively-higher-degree statistics of the training task set. The generalizing solution is approximately a low-degree truncation of the memorizing solution, and so the model first generalizes, then degrades into memorization as training proceeds.

The takeaways are twofold. First, if this case is indicative, learning a "generalizing" solution on an ICL task is really just underfitting the memorizing solution! Second, the fact that kernel machines can generalize on this task means that feature learning is inessential to ICL.

1 INTRODUCTION

Large language models (LLMs) exhibit the striking ability to infer an intended task from context. For example, an LLM told to write a new file in a software repo can match style and content to existing files present in its context. More generally, an LLM given a few examples of a desired task can often infer the task and apply it to new settings. This is called few-shot learning or *in-context learning* (ICL) in the literature. This useful behavior and the conditions under which it arises remain largely mysterious.

How should we begin to describe ICL in mathematical terms? A general setting is as follows. Suppose that there exists a set $\mathcal{B}$ of all possible tasks, of which only a subset $\{\boldsymbol{\beta}^{(t)}\}_{t=1}^{T} =: \mathcal{B}_{\text{tr}} \subset \mathcal{B}$ are present in the model's training data. We might suppose that each task specifies a certain function $f_{\boldsymbol{\beta}} : \mathcal{X} \to \mathcal{Y}$ between two spaces. A given *context* $\boldsymbol{z} \in (\mathcal{X} \times \mathcal{Y})^L \times \mathcal{X}$ consists of a number of examples $(x_i, y_i)_{i=1}^{L}$, where $y_i = f_{\boldsymbol{\beta}}(x_i) + \eta_i$ with noise η_i, together with an unpaired input x_{L+1}. The model is trained to map $\boldsymbol{z} \mapsto f_{\boldsymbol{\beta}}(x_{L+1})$, which requires first inferring $\boldsymbol{\beta}$ from the example pairs and then applying it to the unpaired input x_{L+1}. The most interesting questions arise when the model is provided many examples for each of the T tasks at training time, but at test time is asked to infer a *new* task $\boldsymbol{\beta}_* \notin \mathcal{B}_{\text{tr}}$ from context and apply $f_{\boldsymbol{\beta}_*}$. We would like to know when the model can perform well on unseen tasks — and when it can, to understand why, and how it learned to do so.

It is worth stressing that generalization in ICL is strange indeed, and is in fact rather paradoxical. The Bayes-optimal solution for the training objective is to always interpret any new context $\boldsymbol{z}$ as indicating one of the T training tasks and to simply apply that task. (Or, if there are several plausible candidates, as a distribution over several training tasks.) Given this training objective, it would seem hopeless that the model would correctly infer the unseen task $\boldsymbol{\beta}_* \notin \mathcal{B}_{\text{tr}}$. Nevertheless, it evidently does so in many practical and useful settings. Our work in this paper is to offer a resolution to this paradox and to explain how, in spite of this mismatch, generalization occurs in ICL.

The main idea is that **generalization occurs mid-optimization, as a milestone on the road to memorization.** It is well-appreciated that neural networks learn simple functions early in training and more complex functions later on (e.g. Rahaman et al. (2019); Nakkiran et al. (2019); Xu et al. (2019)). We suggest that the generalizing solution is in fact just a simpler version of the memorizing solution,

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

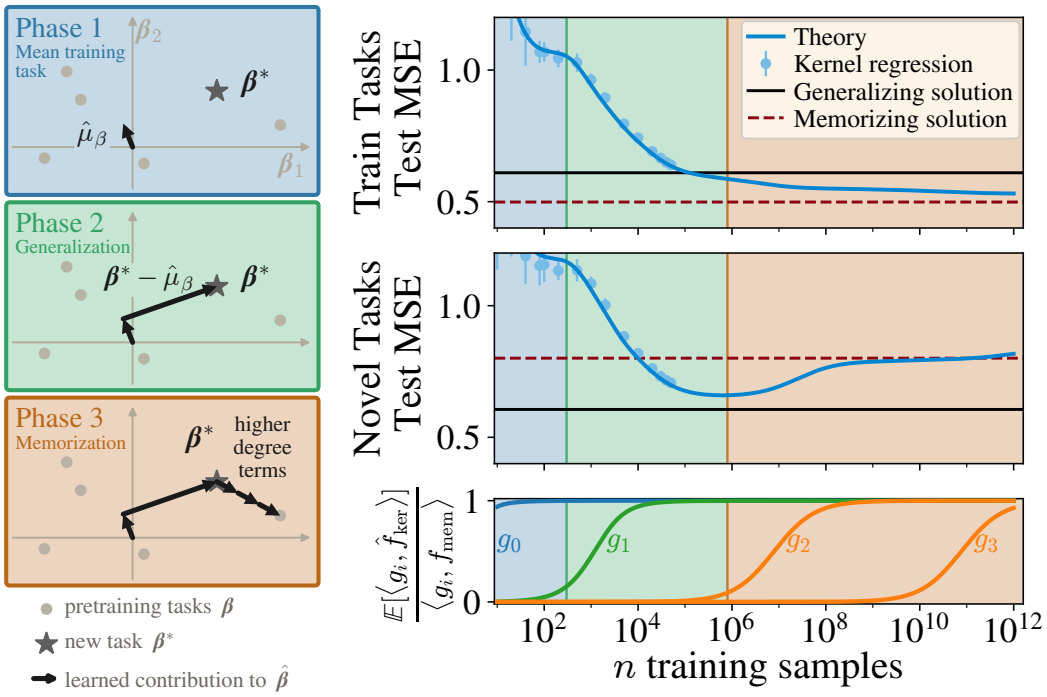


Figure 1: As sample size grows, kernels learn successively higher-degree target function components in in-context linear regression. They first learn the mean training task (phase 1), then approximate the degree-three generalizing solution (phase 2), and finally memorize the training tasks (phase 3). (Left) A visualization of a kernel machine's inferred task in task space $\mathcal{B} = \mathbb{R}^d$ decomposed into vector components learned in each phase (**arrows**). Grey dots indicate training tasks $\mathcal{B}_{\text{tr}} = \{\boldsymbol{\beta}^{(t)}\}$ and a grey star indicates an unseen task $\boldsymbol{\beta}^\star \notin \mathcal{B}_{\text{tr}}$. In phase 1, the kernel learns the mean training task $\hat{\mu}_{\boldsymbol{\beta}} := \mathbb{E}_{\boldsymbol{\beta} \sim \mathcal{B}_{\text{tr}}}[\boldsymbol{\beta}]$. In phase 2, the kernel learns approximately $\boldsymbol{\beta}^\star - \hat{\mu}_{\boldsymbol{\beta}}$, nearly recovering the true task $\boldsymbol{\beta}^\star$. In phase 3, learned higher-degree terms pull the prediction towards the nearest task in $\mathcal{B}_{\text{tr}}$. **(Right)** We plot learning curves for KR on the in-context linear regression problem with $d = 5, L = 7, T = 16$. **(Top)** Evaluated on the training tasks, the error falls monotonically, eventually reaching that of the memorizing solution (Equation (3)). We show both theoretical predictions (lines) and experimental values (dots) in the experimentally-accessible region $n \in [1, 5 \times 10^4]$. **(Middle)** Evaluated on novel tasks ($\boldsymbol{\beta} \sim \mathcal{N}(0, \boldsymbol{I}_d)$), error is minimized at the end of phase 2, approaching that of the generalizing solution (Equation (4)), before rising to that of the memorizing solution in phase 3. Background shading indicates the three phases of learning. **(Bottom)** We also plot the predicted overlap between the function $\hat{f}_{\text{ker}}$ learned by KR and various polynomial functions g_i. The functions g_i correspond to the vector components in the cartoon on the left, and are given explicitly in Section B. The function g_i is determined by the i-th order statistics of $\mathcal{B}_{\text{tr}}$. The component g_2 is responsible for generalization and is learned in phase 2. We show both theoretical predictions (lines) for the kernel's error out to 10^{12} samples, finding excellent agreement between theory and experiment in the experimentally accessible region.gm

stopped before it learns the higher-degree functional components required for exact memorization of the training set.

We arrive at these conclusions from a study of the ICL-for-linear-regression setting of Raventós et al. (2023). In this setting, $\mathcal{X} = \mathbb{R}^d$, $\mathcal{Y} = \mathbb{R}$, $\mathcal{B} = \mathbb{R}^d$, and each task is simply a linear map $f_{\boldsymbol{\beta}}(\boldsymbol{x}) = \boldsymbol{\beta}^\top \boldsymbol{x}$. Task vectors are sampled $\boldsymbol{\beta} \sim \mathcal{N}(0, \boldsymbol{I}_d)$. The beauty of this setting is that the Bayes-optimal train-time (memorizing) and test-time (generalizing) solutions can be written down in closed form (Equations (3) and (4)). Which solution a model learns is largely determined by the model's inductive biases. This suggests studying *kernel regression* (KR) as the model, since there exists a well-developed spectral theory of its inductive bias (Bordelon et al., 2020; Jacot et al., 2020; Simon et al., 2021). The main technical hurdle is in computing the kernel eigendecomposition on this task so that this existing theory may be applied.

Surmounting this hurdle, we find a resolution to the aforementioned paradox: the generalizing solution is approximately a *low-polynomial-degree truncation of the memorizing solution.* As the dataset size grows, kernel machines first learn simple low-degree polynomial components of the memorizing function before resolving increasingly high-degree components. They approximate the generalizing solution after learning the degree-three components of the memorizing solution. Given enough samples, they will proceed to fully memorize. Our results put theory behind the empirics of Raventós et al. (2023), giving the first end-to-end theory of the generalization-memorization transition in ICL. This theory explains the "transience" phenomenon in which a model first learns the generalizing solution before apparently "forgetting it" in favor of the memorizing solution (Singh et al., 2023).

A surprising consequence of our study of fixed-feature kernel machines is that **a model does not need feature learning to do ICL**. (Of course, feature learning certainly allows a model to do ICL more *efficiently.*) This suggests that memorization and generalization in ICL can be understood in simple polynomial language, without reference to feature learning or circuit formation.

2 Research Context and Related Works

Our work sits at the intersection of four threads: the tension between memorizing and generalizing Bayes-optimal solutions (most developed for diffusion models), the dynamics of in-context learning under finite task diversity, simplicity biases in learning, and the eigenstructure theory of kernel regression.

Memorizing and generalizing Bayes-optimal solutions. Whenever a model is trained on a finite sample from some latent distribution, two Bayes-optimal predictors are available: one that treats the empirical distribution as the prior (memorizing), and one that uses the population prior (generalizing). The former generically takes the form of a softmax-weighted average over training points. In our setting, the memorizing solution is the dMMSE (Equation (3)), a posterior average over the T pretraining tasks in *task space*. In diffusion modeling, the memorizing solution is akin to a nearest-neighbor match to the closest training image (Scarvelis et al., 2025; Biroli et al., 2024; Kamb & Ganguli, 2025). In dense associative memories, retrieval is likewise a softmax-weighted average over stored patterns (Krotov & Hopfield, 2016; Ramsauer et al., 2021). The same paradox arises in each case: fully optimizing the training objective gives a predictor that cannot produce anything novel, so any observed generalization must come from the learner *failing* to reach that optimum. The closest analogue to our resolution comes from the diffusion literature. Li et al. (2024) found that linear models capturing the same Gaussian structure as nonlinear diffusion models are nearly identical to the optimal denoiser for Gaussian data. Bonnaire et al. (2025) train a random Fourier features model to show that memorization can be avoided through early stopping, given a strong enough implicit bias. Kamb & Ganguli (2025); Lukoianov et al. (2025) similarly attribute the avoidance of memorization in diffusion models to implicit biases of the network (equivariance) and the data (locality). Together, these results parallel our finding that the generalizing solution is a cumulant truncation of the memorizing one (Proposition 2), implemented through the inductive biases of the learner.

In-context learning. Since its description by Brown et al. (2020), ICL has remained a somewhat enigmatic capability of language models. To understand it, researchers have studied model-side attributes, such as parameter count, depth and the resulting expressivity of the architecture (Brown et al., 2020; Wei et al., 2022b; Garg et al., 2023), as well as data-side attributes, such as task diversity (Chan et al., 2022; Kirsch et al., 2024; Raventós et al., 2023; Park et al., 2025). Closest to our non-attention setting, Tong & Pehlevan (2025) show that MLPs perform ICL linear regression in a similar manner to transformers. On the theoretical side, most analyses of ICL have focused on linear attention (Lu et al., 2025; Letey et al., 2025; Cui et al., 2024; Li et al., 2025). These models are tractable, but they cannot express the memorizing solution, and so cannot capture the transition toward it: they are, by construction, unable to exhibit transience.

Transience. Singh et al. (2023) found that ICL can emerge and later vanish during training. Most explanations of this transience posit competing memorizing and generalizing circuits (Carroll et al., 2025; Singh et al., 2025; Nguyen & Nallamala, 2025). Others (Wurgaft et al., 2025; Park et al., 2025) claim that transience can be explained through a linear combination of the memorizing and generalizing solutions. In Section 6, we explain transience from *first principles* using *fixed-feature* models. To be clear, we give the first description of transience as a consequence of our theory—we

make no curve fits on training trajectories, and we find transience even without models having the ability to learn circuits.

Kernel ridge regression as a model of learning dynamics. Kernel methods have proven useful as models of network dynamics and optimization (Chizat et al., 2019; Du et al., 2019; Bordelon & Pehlevan, 2021), thanks in large part to the neural tangent kernel (NTK) line of work (Neal, 1996; Lee et al., 2018; Jacot et al., 2018), in which suitably initialized wide networks train like kernel learners. Recent works show that the test error of kernel learners can be predicted from the *kernel and task eigenstructure* (Sollich, 2001; Bordelon et al., 2020; Jacot et al., 2020; Simon et al., 2021). This "KRR eigenframework" depends on the kernel's eigenvalues and eigenfunctions with respect to the data distribution, and is reviewed in Section D.

Simplicity biases and statistics of increasing order. Neural networks are known to fit low-frequency or low-complexity functions first (Nakkiran et al., 2019; Abbe et al., 2024). This suggests that the low-to-high ordering of cumulants we find for kernels may also hold for feature-learning networks.

Approximating the kernel eigenstructure. Our eigensystem-finding framework (Definition 4) follows the strategy of Karkada et al. (2026), by approximating the kernel as an inner-product kernel, creating subspaces of polynomials, and performing Gram-Schmidt against the data distribution. Our work trades understandability for expressivity: we cannot generally get closed form expressions for the eigenfunctions, but our framework is more widely applicable.

3 Preliminaries

In-context linear regression. We study the in-context linear regression setting of Raventós et al. (2023), which is defined as follows. At the outset, we construct a pool of task vectors $\mathcal{B}_{\text{tr}} := \{\boldsymbol{\beta}^{(t)}\}_{t=1}^{T}$ of T tasks sampled $\boldsymbol{\beta}^{(t)} \sim \mathcal{N}(0, \boldsymbol{I}_d)$ to generate the training data. In this setting, one training sample is a *context vector* constructed as follows. First, a task $\boldsymbol{\beta}$ is drawn uniformly from $\mathcal{B}_{\text{tr}}$. Second, $L+1$ vectors $(\boldsymbol{x}_i)_{i=1}^{L+1}$ are sampled $\boldsymbol{x}_i \sim \mathcal{N}(0, \frac{1}{d}\boldsymbol{I}_d)$.[1] Third, $L+1$ values $(y_i)_{i=1}^{L+1}$ are computed as $y_i = \boldsymbol{\beta}^\top \boldsymbol{x}_i + \eta_i$, with noise $\eta_i \sim \mathcal{N}(0, \epsilon^2)$. Finally, we construct the context

$$\boldsymbol{z} = \left[\boldsymbol{x}_1{}^\top, y_1, \ldots, \boldsymbol{x}_L{}^\top, y_L, \boldsymbol{x}_{L+1}{}^\top\right] \in \mathbb{R}^{(d+1)L+d}, \tag{1}$$

with $y_{L+1} =: y$ held out as the regression target. We will write $\mathcal{D}_{\boldsymbol{\beta}}$ for the joint distribution of $(\boldsymbol{z}, y)$ conditioned on a given task vector $\boldsymbol{\beta}$. We refer to T as the *task diversity* and L as the *context length*.

Kernel regression. We work in a standard supervised setting in which the model is trained on N contexts $\mathcal{Z} = \left\{\boldsymbol{z}^{(n)}\right\}_{n=1}^{N}$ and the corresponding scalar targets $\boldsymbol{y} = \left(y^{(n)}\right)_{n=1}^{N}$. We take KR as the learning algorithm. KR is specified by a choice of positive-semidefinite kernel function $K(\boldsymbol{z}, \boldsymbol{z}')$ and a choice of ridge parameter $\delta \geq 0$. We consider various rotation-invariant kernel functions: $K(\boldsymbol{z}, \boldsymbol{z}') = K(\boldsymbol{z}^\top \boldsymbol{z}', \|\boldsymbol{z}\|, \|\boldsymbol{z}'\|)$. KR returns the predicted function $\hat{f}(\boldsymbol{z}) = K_{\boldsymbol{z}\mathcal{Z}} \left(K_{\mathcal{Z}\mathcal{Z}} + \delta \boldsymbol{I}_N\right)^{-1} \boldsymbol{y}$, where $[K_{\boldsymbol{z}\mathcal{Z}}]_m = K(\boldsymbol{z}, \boldsymbol{z}^{(n)})$ and $[K_{\mathcal{Z}\mathcal{Z}}]_{mn} = K(\boldsymbol{z}^{(m)}, \boldsymbol{z}^{(n)})$.

Evaluating the learned function. Once we have the learned function $\hat{f}$, this will evaluated two ways, on two distributions. First, we study its error on new contexts from the *training task distribution:* $\mathcal{D}_{\text{tr}} := \frac{1}{T}\sum_{\boldsymbol{\beta} \in \mathcal{B}_{\text{tr}}} \mathcal{D}_{\boldsymbol{\beta}}$. Second, we compute its error on the *prior distribution*, with contexts generated from fresh tasks: $\mathcal{D}_* := \mathbb{E}_{\boldsymbol{\beta} \sim \mathcal{N}(0, \boldsymbol{I}_d)}[\mathcal{D}_{\boldsymbol{\beta}}]$. We compute mean-squared error (MSE) over both distributions:

$$\mathcal{E}_{\text{tr}}(\hat{f}) := \mathop{\mathbb{E}}_{(\boldsymbol{z}, y) \sim \mathcal{D}_{\text{tr}}} \left[\left(\hat{f}(\boldsymbol{z}) - y\right)^2\right], \qquad \mathcal{E}_*(\hat{f}) := \mathop{\mathbb{E}}_{(\boldsymbol{z}, y) \sim \mathcal{D}_*} \left[\left(\hat{f}(\boldsymbol{z}) - y\right)^2\right]. \tag{2}$$

Memorizing and generalizing solutions The *memorizing solution* is defined as that which minimizes error the training distribution: $f_{\text{mem}} := \operatorname{argmin}_f \mathcal{E}_{\text{tr}}(f)$. As shown by Raventós et al. (2023), f_{mem} takes the discrete-minimum-mean-squared-error (dMMSE) form

$$f_{\text{mem}}(\boldsymbol{z}) = \frac{\sum_{t=1}^{T} w_t(\boldsymbol{z}) \cdot \boldsymbol{\beta}^{(t)\top} \boldsymbol{x}_{L+1}}{\sum_{t=1}^{T} w_t(\boldsymbol{z})}, \quad \text{where} \quad w_t = \exp\left(-\frac{1}{2\epsilon^2}\sum_{i=1}^{L}\left(y_i - \boldsymbol{\beta}^{(t)\top}\boldsymbol{x}_i\right)^2\right). \tag{3}$$

[1] Anisotropic distributions for $\boldsymbol{x}$ are discussed in Sections E, F and H.

The *generalizing solution* is the Bayes-optimal solution on the prior distribution: $f_{\text{gen}} := \text{argmin}_f \mathcal{E}_*(f)$. This is simply ridge regression on the context:

$$f_{\text{gen}}(\boldsymbol{z}) = \boldsymbol{x}_{L+1}^T \boldsymbol{X} \left(\boldsymbol{X}^\top \boldsymbol{X} + \epsilon^2 \boldsymbol{I}_d \right)^{-1} \boldsymbol{y}_{1:L}, \qquad \text{where } \boldsymbol{X} = [\boldsymbol{x}_1, \cdots, \boldsymbol{x}_L]. \tag{4}$$

We will find that, as sample size N grows, KR first learns f_{gen}, then learns f_{mem} as $N \to \infty$.

Moments and cumulants. We write $\hat{\boldsymbol{\mu}}_{\boldsymbol{\beta}} := \frac{1}{T} \sum_t \beta^{(t)}$ for the mean of the training tasks $\mathcal{B}_{\text{tr}}$ and $\hat{\Sigma}_{\boldsymbol{\beta}}$ for its covariance. We write $\hat{\kappa}_\ell$ for the order-ℓ cumulant tensor of $\mathcal{B}_{\text{tr}}$.

4 The Kernel Eigenstructure

4.1 Review of the KR Eigenframework

Prior work has obtained an predictive analytic framework which gives the generalization behavior of KR on essentially arbitrary distributions (see e.g. Simon et al. (2021)). This *KR eigenframework* takes as inputs the kernel eigensystem and the associated eigendecomposition of the target function. The eigensystem of a kernel function $K(\cdot, \cdot)$ w.r.t. a distribution $\mathcal{D}$ is the set of scalar-function pairs λ_i, ϕ_i that satisfy $\mathbb{E}_{\boldsymbol{z}' \sim \mathcal{D}}[K(\boldsymbol{z}, \boldsymbol{z}')\phi_i] = \lambda_i \phi_i$. A target function f_* is then decomposed into coefficients $v_i = \langle \phi_i, f_* \rangle_{\mathcal{D}}$. The KR eigenframework gives equations predicting quantities such as $\mathbb{E}\left[\hat{f}\right]$ and the test-time MSE. The main qualitative idea is that the target components $v_i \phi_i$ are learned in decreasing order of λ_i: eigenfunctions with larger eigenvalue are learned first. This behavior is known as *spectral bias.*

4.2 The Kernel Eigensystem in In-Context Linear Regression

In order to use the KR eigenframework in our ICL setting, we need to compute the task eigensystem and target eigencoefficients (or at least say enough to draw useful conclusions). The training data are sampled $\boldsymbol{z} \sim \mathcal{D}_{\text{tr}}$ and have target function $\mathbb{E}[y|\boldsymbol{z}] = f_{\text{mem}}(\boldsymbol{z})$. We thus need to eigendecompose a rotation-invariant kernel K w.r.t. distribution $\mathcal{D}_{\text{tr}}$.

This eigensystem cannot generally be found in closed form, so to make progress, we invoke a series of assumptions and approximations. We give a sketch here and a derivation in Section E. First, following Karkada et al. (2026), we approximate the kernel as an *inner-product kernel* $\tilde{K}(\boldsymbol{z}, \boldsymbol{z}') = \sum_{\ell \geq 0} \frac{c_\ell}{\ell!} (\boldsymbol{z}^\top \boldsymbol{z}')^\ell$, obtaining the constants $\{c_\ell\}$ by matching K and $\tilde{K}$ on the sphere $r \cdot \mathbb{S}^{\dim(\boldsymbol{z})-1}$ with $r = \text{rms}\left[\|\boldsymbol{z}\|\right]$. Then, as in that same paper, we assume $c_\ell \gg c_{\ell+1}$, which implies that the kernel eigensystem can be found by:

1. taking the subspaces $(\mathcal{P}_{\leq \ell})_{\ell=0}^{\infty}$, where $\mathcal{P}_{\leq \ell}$ contains all polynomials of $\boldsymbol{z}$ up to order ℓ;
2. performing Gram-Schmidt on these subspaces in order of increasing ℓ to obtain orthogonal subspaces $(\tilde{\mathcal{P}}_{\leq \ell})_{\ell=0}^{\infty}$; and finally
3. diagonalizing each subspace $\tilde{\mathcal{P}}_{\leq \ell}$ w.r.t. $\mathcal{D}_{\text{tr}}$.

Within orders $\ell = 0 \ldots 4$, some of the eigenfunctions can be found in closed form (and the rest may be found numerically).[2] The two most important (sets of) eigenfunctions that emerge analytically in this way are as follows:

- at order one, the d "linear readout" modes $\boldsymbol{\phi}_1(\boldsymbol{z}) = \boldsymbol{x}_{L+1}$, which have eigenvalue $\lambda_1 = \frac{c_1}{d}$; and
- at order three, the d "regression" components

$$\boldsymbol{\phi}_3(\boldsymbol{z}) \approx \left(\sum_{i=1}^{L} y_i \boldsymbol{x}_i^\top - \frac{L}{d} \hat{\boldsymbol{\mu}}_{\boldsymbol{\beta}}^\top \right) \boldsymbol{x}_{L+1}, \tag{5}$$

  which have eigenvalue $\lambda_3 \propto L \frac{c_3}{d^3}$ ($c_3/c_1 \sim L^{-1}$).

[2] The fact that no eigenfunctions of order $\ell \geq 5$ can be found analytically is a consequence of the insolubility of the quintic!

In general, the top eigenvalues at order ℓ scale like $c_\ell \cdot L^{(\ell-1)/2} \cdot d^{-\ell}$ (c_ℓ scaling as $L^{-\ell/2}$) and thus dominate those of order $\ell+1$, and thus the levels are learned in sequence.

We experimentally check our eigensystem against a true numerical eigensystem computed from a finite trainset for several common rotation-invariant kernels in Section E. We see very good numerical agreement, which validates the assumptions we have made up to this point. In particular, it confirms that these kernels will indeed learn target function components in order of increasing degree.

4.3 Eigendecomposing the generalizing and memorizing solutions

We must now find the eigencoefficients of f_{gen} and f_{mem}. We again sketch our results here and defer a full treatment to Section F. We first observe that as the empirical $\boldsymbol{x}$-covariance $\boldsymbol{X}^\top \boldsymbol{X} \to \text{const} \cdot \boldsymbol{I}_d$, then f_{gen} approaches the third-order polynomial $\boldsymbol{x}_{L+1}^\top \boldsymbol{X} \boldsymbol{y}_{1:L}$. This polynomial is the primary component in $\boldsymbol{\phi}_3$ (and in any case is fully contained in $\bigcup_{\ell \le 3} \tilde{P}_{\le \ell}$). Indeed, $\boldsymbol{X}^\top \boldsymbol{X} \approx \text{const} \cdot \boldsymbol{I}_d$ so long as $L \gg d$ (which is indeed the regime we work in).

We can analytically decompose both f_{mem} up to order 3. Truncating f_{mem} at order 3, the function approximates the exact same third-degree polynomial as f_{gen}. Thus, f_{gen} is approximately a degree-3 polynomial truncation of f_{mem}, and will be learned by the kernel with approximately $L^2 d^2/2$ samples. Past the degree-3 truncation, f_{mem} contains higher-degree polynomials that do not exist in f_{gen}.

By symmetry, all even-order eigencoefficients of both f_{mem} and f_{gen} are zero.

4.4 The resulting picture of memorization and generalization

As the training sample size N grows, KR will learn successively lower-eigenvalue (i.e. higher-degree) components of f_{mem}, on which it is trained. Once it has learned the third-order components of f_{mem}, its learned function $\hat{f}$ in fact approximates f_{gen}. As N grows further, $\hat{f}$ picks up yet higher-order components of f_{mem}, eventually arriving at the memorizing solution. In Figure 5, we show how a model trained without feature learning can approximate f_{mem}. In Figures 2 and 4, we experimentally show the same models approaching f_{gen} once the degree-3 polynomial is expressed, and strays from f_{gen} once the higher degree terms present in f_{mem} alone are implemented.

5 The learning dynamics in task space

Although we cannot solve for the eigenfunctions in closed form for every degree, and thus cannot find the eigencoefficients of f_{mem} in closed form, there is still regularity in the eigendecomposition of f_{mem}. As f_{mem} is linear in $\boldsymbol{x}_{L+1}$, every eigenfunction with a nonzero eigencoefficient must also be linear in $\boldsymbol{x}_{L+1}$, the other odd terms having been Gram-Schmidt against the linear term. That implies that the kernel prediction itself can be written as a linear function of $\boldsymbol{x}_{L+1}$: $\hat{f}_{\text{ker}} = \hat{\boldsymbol{\beta}}^\top \boldsymbol{x}_{L+1}$. We can use understand the kernel's predictions not just in terms of the eigenfunctions, but also in the intuitive *task space* $\mathcal{B}$ itself!

5.1 The generalizing and memorizing solutions in task space

Similar to Section 4, we are able to describe $f_{\text{mem}} = \boldsymbol{\beta}_{\text{mem}}^\top \boldsymbol{x}_{L+1}$ as a truncation of $f_{\text{gen}} = \boldsymbol{\beta}_{\text{gen}}^\top \boldsymbol{x}_{L+1}$, but now in the task space. We denote the true task we wish to estimate as $\boldsymbol{\beta}^\star$. Note that $\boldsymbol{\beta}_{\text{gen}} \approx \boldsymbol{\beta}^\star$ by construction, so we should find that the low degree truncation of $\boldsymbol{\beta}_{\text{mem}} \approx \boldsymbol{\beta}^\star$ as well.

To continue, we note that $\boldsymbol{\beta}_{\text{mem}}(\boldsymbol{z}) \approx \boldsymbol{v}_1 + \boldsymbol{v}_3 \left(\sum_i y_i \boldsymbol{x}_i - \frac{L}{d} \hat{\mu}_{\boldsymbol{\beta}} \right)$. $\boldsymbol{v}_1$ can be exactly found using a simple argument: if the training tasks $\mathcal{B}_{\text{tr}}$ were highly concentrated around some mean $\hat{\boldsymbol{\mu}}_{\boldsymbol{\beta}}$, the Bayes-optimal solution given no contextual information would be to guess $\hat{\boldsymbol{\mu}}_{\boldsymbol{\beta}}$. As we confirm in Table 2, $\boldsymbol{v}_1$ is indeed $\hat{\boldsymbol{\mu}}_{\boldsymbol{\beta}}$.

$\boldsymbol{\beta}_{\text{gen}}$ does not use $\boldsymbol{\mu}_{\boldsymbol{\beta}}$ (as we set the mean to be $\boldsymbol{0}$), so what is happening here? Does this mean f_{gen} isn't a truncation of f_{mem} after all? $\boldsymbol{\beta}_{\text{mem}}$ using $\hat{\boldsymbol{\mu}}_{\boldsymbol{\beta}}$ for its first term is perfectly fine, as long as we subtract it when we implement the remaining term. Evaluating it, we have $\sum_i y_i \boldsymbol{x}_i - \frac{L}{d} \hat{\boldsymbol{\mu}}_{\boldsymbol{\beta}} \approx \frac{L}{d} (\boldsymbol{\beta}^\star - \hat{\boldsymbol{\mu}}_{\boldsymbol{\beta}})$, with $\boldsymbol{v}_3 = \boldsymbol{1}$ (derived in Section G). Training a kernel, the predicted task would thus begin by predicting the average task, then this average task would be forgotten in favor of approximating the true task! Trained on more data, a kernel incorporates higher order functions of $\boldsymbol{\beta}^\star$ that purely memorizing.

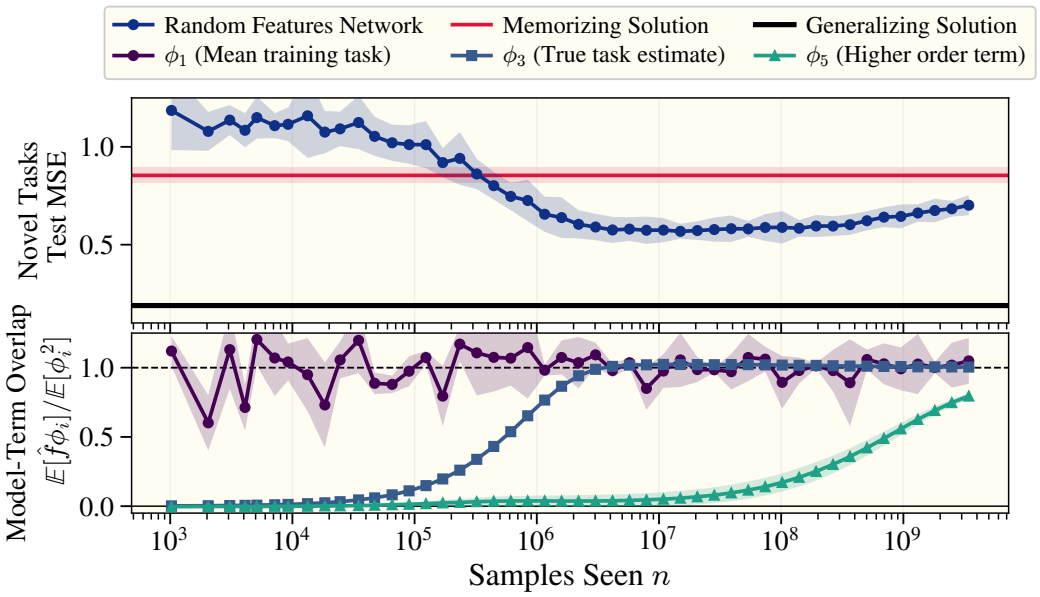


Figure 2: Transience is explained by learning higher order cumulants. We train random features networks (see Section B.1 for a discussion of random features networks and why we use them) is trained with online SGD on ICL linear regression data. **(Top.)** Test MSE on novel tasks $\beta \sim \mathcal{B}_{\text{prior}}$, with the memorizing (dMMSE, red) and generalizing (ridge, black) oracles for reference. **(Bottom.)** Overlap of the learned predictor with the leading terms of the memorizing solution, $\mathbb{E}[\hat{f}\phi_i]/\mathbb{E}[\phi_i^2]$, for the degree-1 term $\boldsymbol{\phi}_1 = \boldsymbol{x}_{L+1}$ (with eigencoefficient of the pool mean), the degree-3 term $\phi_3 \approx \left(\sum_i y_i \boldsymbol{x}_i^\top - \frac{L}{d}\hat{\boldsymbol{\mu}}_{\boldsymbol{\beta}}^\top\right)\boldsymbol{x}_{L+1}$ (with an eigencoefficient that's approximately 1), and the degree-5 term $\boldsymbol{\phi}_5 \sim \sum_i y_i y_j \boldsymbol{x}_i^\top \boldsymbol{x}_j \boldsymbol{x}_{L+1}$ at leading order (the eigencoefficient depends on the training pool's skewness; see Theorem 1). Novel-task error drops below the memorizing oracle as ϕ_3 is learned ($n \approx 10^5$–10^6) and is lowest once ϕ_3 saturates. It then rises back toward the memorizing oracle as the network incorporates the training task distribution's skewness through ϕ_5 ($n \gtrsim 10^7$). Further experimental details are in Section B.

5.2 The generalizing and memorizing picture in task space

The dynamical picture in the task space is given in Figure 1 (left), where a kernel first learns the training task's mean. During the generalizing phase, the kernel approximates $\boldsymbol{\beta}^\star$ after subtracting the initial guess of $\hat{\boldsymbol{\mu}}_{\boldsymbol{\beta}}$. Finally, highly nonlinear terms of $\boldsymbol{\beta}^\star$ are learned, eventually converging to the memorizing solution.

6 Transience is an underfitting phenomenon

In terms of both eigenfunctions of the data (Section 4) and the approximated task in task space (Section 5), we have found f_{gen} to approximately be a low-degree truncation of f_{mem}. With this fact, we are now well-positioned to understand the "transience of ICL" phenomenon, wherein models first generalize before they memorize.

For models that implement simple functions early in training (see Nakkiran et al. (2019) for relevant models), we can attribute transience to these models learning the low-degree truncation of the memorizing solution as opposed to implementing the generalizing solution. Learning more complex functions later in training can thus be understood as the model memorizing. We experimentally validate our theory in Figure 2 by using random feature networks (discussed in Section B), which serve as sample-efficient approximations of kernels.

Our simple picture allows us to resolve the paradox we initially posed: why can a model generalize even though it is optimized solely using the training objective with finite training tasks?

- Models *are always* minimizing the training objective.

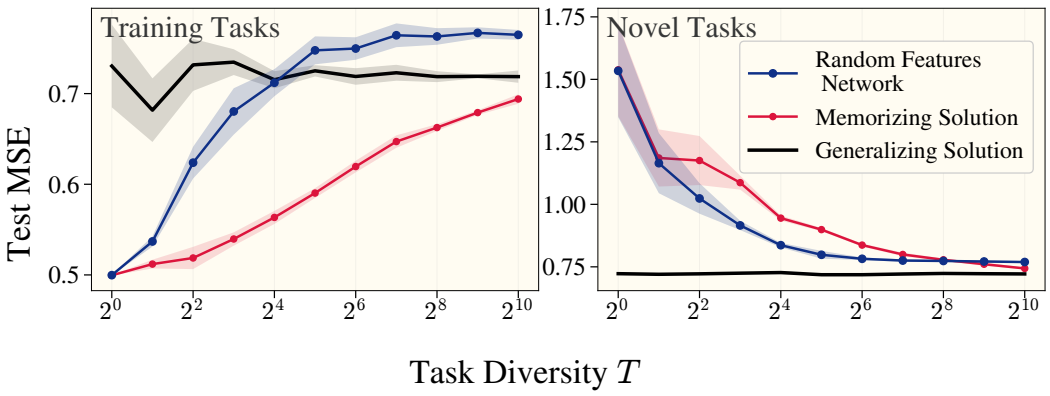


Figure 3: Task diversity trades pretraining-task accuracy for novel-task accuracy. Following Raventós et al. (2023), we sweep the number of training tasks T and report test MSE on the pretraining tasks $\mathcal{B}_{\text{tr}}$ (**left**) and on novel tasks $\mathcal{B}_{\text{prior}}$ (**right**). Our data is drawn with $d = 8, L = 15, \epsilon^2 = 0.25$ In blue, we show a fit random features network trained with SGD (see Section B for random feature network details), with the memorizing (dMMSE) and generalizing (**ridge**) oracles shown. As T grows, the training task distribution's cumulants approach that of the prior's and all three curves converge. Evaluated on novel tasks, the random features model initially tracks the generalizing solution up to a task diversity T of 4, continuing to approach the generalizing solution until $T \approx 2^6 = 32$ tasks. The model does not exactly reach the generalizing solution due to the finite context length L we use for training. Further experimental details are in Section B.

- While minimizing this training objective, they implement the simple solution of a low-degree truncation of the memorizing solution.
- This low-degree truncation is approximately the generalizing solution.

7 The Effect of Task Diversity

Given models learn an approximation of the generalizing solution by implementing a low-degree truncation of the memorizing solution, how does this compare to the task diversity picture in which models trained with higher task diversity generalize (Raventós et al., 2023)?

Our findings are perfectly compatible with this line of research. As the task diversity increases, the mean and covariance (along with the higher order cumulants) of the training distribution $\mathcal{B}_{\text{tr}}$ become closer to those of the prior distribution $\mathcal{B}_{\text{prior}}$. The mean and covariance specify the memorizing solution's optimal polynomial degree 3 truncation; better estimating the Gaussian statistics directly impacts the quality of the truncated f_{mem} being an approximation of f_{gen}. To directly show this, in Figure 5, we take exactly $T = 4$ tasks on $d = 3$ dimensional data, whiten the covariance, and find a random feature model closely approaches f_{gen}.

We substantiate our theories compatibility with the task diversity picture experimentally in Figure 3, where we train a random feature model. We find that, evaluated on a distribution with novel tasks, our model that cannot feature learn tracks the f_{mem} for low task diversity T, and approaches f_{gen} at high task diversity T.

8 Further Discussion

We have given our headline result, that feature learning is not needed to perform in-context learning, and provided a first-principles account of why. Within that scope, we have left out two smaller findings that deserve a brief discussion.

- Should we drop the assumption that our in-context data $\boldsymbol{x}$ is drawn isotropically (as is done in Section F), we find that the eigencoefficients will be determined not using the cumulants of $\mathcal{B}_{\text{tr}}$, but of a colored version with using the cumulants of the uniform distribution of $\{\Sigma_{\boldsymbol{x}}^{1/2}\boldsymbol{\beta}^{(t)}\}_{t=1}^{T}$.

- A model implementing the optimal polynomial degree 3 truncation of f_{mem} will achieve its lowest loss on novel tasks; implementing higher degree terms can only hurt performance on novel tasks.

To end, we hope this work shows the usefulness of solvable models to advance our understanding of seemingly nebulous phenomena.

Statement on the Use of Large Language Models

We used large language models (LLMs) for writing code, giving a precise expansion of the Ridge and dMMSE functions in terms of cumulants and drafting their appendices, refining the mathematics of high order subkernel diagonalization, helping clarify the position of related works, and constructing proofs. As LLMs often made various implicit assumptions about the structure of the empirical cumulants, we independently verified all outputs of LLMs.

REFERENCES

Emmanuel Abbe, Enric Boix-Adsera, and Theodor Misiakiewicz. The merged-staircase property: a necessary and nearly sufficient condition for sgd learning of sparse functions on two-layer neural networks, 2024. URL .

Francis Bach. High-dimensional analysis of double descent for linear regression with random projections. *arXiv preprint arXiv:2303.01372*, 2023.

Giulio Biroli, Tony Bonnaire, Valentin de Bortoli, and Marc Mézard. Dynamical regimes of diffusion models. *Nature Communications*, 15(1), November 2024. ISSN 2041-1723. doi: 10.1038/s41467-024-54281-3. URL .

Tony Bonnaire, Raphaël Urfin, Giulio Biroli, and Marc Mézard. Why diffusion models don't memorize: The role of implicit dynamical regularization in training, 2025. URL .

Blake Bordelon and Cengiz Pehlevan. Learning curves for sgd on structured features. *arXiv preprint arXiv:2106.02713*, 2021.

Blake Bordelon, Abdulkadir Canatar, and Cengiz Pehlevan. Spectrum dependent learning curves in kernel regression and wide neural networks. In *International Conference on Machine Learning*, pp. 1024–1034. PMLR, 2020.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

Abdulkadir Canatar, Blake Bordelon, and Cengiz Pehlevan. Spectral bias and task-model alignment explain generalization in kernel regression and infinitely wide neural networks. *Nature communications*, 12(1):1–12, 2021.

Abdulkadir Canatar, Blake Bordelon, and Cengiz Pehlevan. Out-of-distribution generalization in kernel regression, 2022. URL .

Andrea Caponnetto and Ernesto De Vito. Optimal rates for the regularized least-squares algorithm. *Foundations of Computational Mathematics*, 7(3):331–368, 2007.

Liam Carroll, Jesse Hoogland, Matthew Farrugia-Roberts, and Daniel Murfet. Dynamics of transient structure in in-context linear regression transformers, 2025. URL .

Stephanie C. Y. Chan, Adam Santoro, Andrew K. Lampinen, Jane X. Wang, Aaditya Singh, Pierre H. Richemond, Jay McClelland, and Felix Hill. Data distributional properties drive emergent in-context learning in transformers, 2022. URL .

Chen Cheng and Andrea Montanari. Dimension free ridge regression. *arXiv preprint arXiv:2210.08571*, 2022.

Lenaic Chizat, Edouard Oyallon, and Francis Bach. On lazy training in differentiable programming. *Advances in Neural Information Processing Systems*, 32, 2019.

Hugo Cui, Freya Behrens, Florent Krzakala, and Lenka Zdeborová. A phase transition between positional and semantic learning in a solvable model of dot-product attention. In *Advances in Neural Information Processing Systems 37*, NeurIPS 2024, pp. 36342–36389. Neural Information Processing Systems Foundation, Inc. (NeurIPS), 2024. doi: 10.52202/079017-1146. URL .

Edgar Dobriban and Stefan Wager. High-dimensional asymptotics of predictions: Ridge regression and classification. *The Annals of Statistics*, 46(1):247–279, 2018.

Simon Du, Jason Lee, Haochuan Li, Liwei Wang, and Xiyu Zhai. Gradient descent finds global minima of deep neural networks. In *International Conference on Machine Learning*, pp. 1675–1685. PMLR, 2019.

Shivam Garg, Dimitris Tsipras, Percy Liang, and Gregory Valiant. What can transformers learn in-context? a case study of simple function classes, 2023. URL .

Trevor Hastie, Andrea Montanari, Saharon Rosset, and Ryan J Tibshirani. Surprises in high-dimensional ridgeless least squares interpolation. *The Annals of Statistics*, 50(2):949–986, 2022.

Arthur Jacot, Clément Hongler, and Franck Gabriel. Neural tangent kernel: Convergence and generalization in neural networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2018.

Arthur Jacot, Berfin Şimşek, Francesco Spadaro, Clément Hongler, and Franck Gabriel. Kernel alignment risk estimator: Risk prediction from training data. *Advances in neural information processing systems*, 33:15568–15578, 2020.

Mason Kamb and Surya Ganguli. An analytic theory of creativity in convolutional diffusion models, 2025. URL .

Dhruva Karkada, Joseph Turnbull, Yuxi Liu, and James B. Simon. Predicting kernel regression learning curves from only raw data statistics, 2026. URL .

Louis Kirsch, James Harrison, Jascha Sohl-Dickstein, and Luke Metz. General-purpose in-context learning by meta-learning transformers, 2024. URL .

Dmitry Krotov and John J Hopfield. Dense associative memory for pattern recognition, 2016. URL .

Jaehoon Lee, Yasaman Bahri, Roman Novak, Samuel S. Schoenholz, Jeffrey Pennington, and Jascha Sohl-Dickstein. Deep neural networks as gaussian processes. In *International Conference on Learning Representations (ICLR)*, 2018.

Mary I. Letey, Jacob A. Zavatone-Veth, Yue M. Lu, and Cengiz Pehlevan. Pretrain-test task alignment governs generalization in in-context learning, 2025. URL .

Xiang Li, Yixiang Dai, and Qing Qu. Understanding generalizability of diffusion models requires rethinking the hidden gaussian structure, 2024. URL .

Yingcong Li, Davoud Ataee Tarzanagh, Ankit Singh Rawat, Maryam Fazel, and Samet Oymak. Gating is weighting: Understanding gated linear attention through in-context learning, 2025. URL .

Bruno Loureiro, Cedric Gerbelot, Hugo Cui, Sebastian Goldt, Florent Krzakala, Marc Mezard, and Lenka Zdeborová. Learning curves of generic features maps for realistic datasets with a teacher-student model. *Advances in Neural Information Processing Systems*, 34:18137–18151, 2021.

Yue M. Lu, Mary Letey, Jacob A. Zavatone-Veth, Anindita Maiti, and Cengiz Pehlevan. Asymptotic theory of in-context learning by linear attention. *Proceedings of the National Academy of Sciences*, 122(28), 2025. ISSN 1091-6490. doi: 10.1073/pnas.2502599122. URL .

Artem Lukoianov, Chenyang Yuan, Justin Solomon, and Vincent Sitzmann. Locality in image diffusion models emerges from data statistics, 2025. URL .

Preetum Nakkiran, Gal Kaplun, Dimitris Kalimeris, Tristan Yang, Benjamin L. Edelman, Fred Zhang, and Boaz Barak. Sgd on neural networks learns functions of increasing complexity, 2019. URL .

Radford M Neal. Priors for infinite networks. In *Bayesian Learning for Neural Networks*, pp. 29–53. Springer, 1996.

Alex Nguyen and Gautam Reddy Nallamala. Differential learning kinetics govern the transition from memorization to generalization during in-context learning. In Y. Yue, A. Garg, N. Peng, F. Sha, and R. Yu (eds.), *International Conference on Learning Representations*, volume 2025, pp. 8677–8697, 2025. URL .

Core Francisco Park, Ekdeep Singh Lubana, Itamar Pres, and Hidenori Tanaka. Competition dynamics shape algorithmic phases of in-context learning, 2025. URL .

Nasim Rahaman, Aristide Baratin, Devansh Arpit, Felix Draxler, Min Lin, Fred Hamprecht, Yoshua Bengio, and Aaron Courville. On the spectral bias of neural networks. In *International Conference on Machine Learning*, pp. 5301–5310. PMLR, 2019.

Ali Rahimi and Benjamin Recht. Random features for large-scale kernel machines. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 3, pp. 5, 2007.

Hubert Ramsauer, Bernhard Schäfl, Johannes Lehner, Philipp Seidl, Michael Widrich, Thomas Adler, Lukas Gruber, Markus Holzleitner, Milena Pavlović, Geir Kjetil Sandve, Victor Greiff, David Kreil, Michael Kopp, Günter Klambauer, Johannes Brandstetter, and Sepp Hochreiter. Hopfield networks is all you need, 2021. URL .

Allan Raventós, Mansheej Paul, Feng Chen, and Surya Ganguli. Pretraining task diversity and the emergence of non-bayesian in-context learning for regression, 2023. URL .

Dominic Richards, Jaouad Mourtada, and Lorenzo Rosasco. Asymptotics of ridge (less) regression under general source condition. In *International Conference on Artificial Intelligence and Statistics*, pp. 3889–3897. PMLR, 2021.

Christopher Scarvelis, Haitz Sáez de Ocáriz Borde, and Justin Solomon. Closed-form diffusion models, 2025. URL .

James B Simon, Madeline Dickens, Dhruva Karkada, and Michael R DeWeese. The eigenlearning framework: A conservation law perspective on kernel ridge regression and wide neural networks. *arXiv preprint arXiv:2110.03922*, 2021.

Aaditya K. Singh, Stephanie C. Y. Chan, Ted Moskovitz, Erin Grant, Andrew M. Saxe, and Felix Hill. The transient nature of emergent in-context learning in transformers, 2023. URL .

Aaditya K. Singh, Ted Moskovitz, Sara Dragutinovic, Felix Hill, Stephanie C. Y. Chan, and Andrew M. Saxe. Strategy coopetition explains the emergence and transience of in-context learning, 2025. URL .

Peter Sollich. Gaussian process regression with mismatched models. In *Advances in Neural Information Processing Systems*, pp. 519–526. MIT Press, 2001.

William L. Tong and Cengiz Pehlevan. Mlps learn in-context on regression and classification tasks, 2025. URL .

Alexander Wei, Wei Hu, and Jacob Steinhardt. More than a toy: Random matrix models predict how real-world neural representations generalize. In *International Conference on Machine Learning*, Proceedings of Machine Learning Research, 2022a.

Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, Ed H. Chi, Tatsunori Hashimoto, Oriol Vinyals, Percy Liang, Jeff Dean, and William Fedus. Emergent abilities of large language models, 2022b. URL .

Denny Wu and Ji Xu. On the optimal weighted ℓ_2 regularization in overparameterized linear regression. *Advances in Neural Information Processing Systems*, 33:10112–10123, 2020.

Daniel Wurgaft, Ekdeep Singh Lubana, Core Francisco Park, Hidenori Tanaka, Gautam Reddy, and Noah D. Goodman. In-context learning strategies emerge rationally, 2025. URL .

Zhi-Qin John Xu, Yaoyu Zhang, and Yanyang Xiao. Training behavior of deep neural network in frequency domain. In *International Conference on Neural Information Processing*, pp. 264–274. Springer, 2019.

A Review of Notation

As notation in ICL literature evolves constantly, we provide a table to reference all important quantities.

Symbol	Meaning
	Data and task distribution
d	In-context data dimension
L	Context length
T	Number of pretraining tasks
n	Number of training samples (contexts) supplied to the kernel
$\boldsymbol{\beta}$	General notation for tasks
$\mathcal{B}_{\text{prior}} = \mathcal{N}(\mu_{\boldsymbol{\beta}}, \Sigma_{\boldsymbol{\beta}})$	Prior distribution used to draw tasks
$\mathcal{B}_{\text{tr}} = \{\boldsymbol{\beta}^{(t)}\}_{t=1}^{T}$	Training pool of tasks
$\boldsymbol{z}$	Context (samples) consisting of $(\boldsymbol{x}_1^\top, y_1, \ldots, \boldsymbol{x}_L^\top, y_L, \boldsymbol{x}_{L+1}^\top)$
$\Sigma_{\boldsymbol{x}}$	Covariance of the in-context data $\boldsymbol{x}_i$
ϵ^2	Label noise variance, $y_i = \boldsymbol{\beta}^\top \boldsymbol{x}_i + \epsilon\eta_i$
δ	Ridge parameter in KRR
$\hat{\boldsymbol{\mu}}_{\boldsymbol{\beta}},\ \hat{\Sigma}_{\boldsymbol{\beta}}$	Mean and covariance of $\mathcal{B}_{\text{tr}}$ (i.e. $\hat{\kappa}_1, \hat{\kappa}_2$)
$\kappa_\ell,\ \hat{\kappa}_\ell$	Order-ℓ cumulant of $\mathcal{B}_{\text{prior}}$ and $\mathcal{B}_{\text{tr}}$, respectively
M_m	Order-m raw moment tensor of $\mathcal{B}_{\text{tr}}$
	Whitened quantities (Given $\Sigma_{\boldsymbol{x}} \neq cI_d$)
$\tilde{\boldsymbol{x}} = \Sigma_{\boldsymbol{x}}^{-1/2}\boldsymbol{x}$	Whitened in-context data
$\tilde{\boldsymbol{\beta}} = \Sigma_{\boldsymbol{x}}^{1/2}\boldsymbol{\beta}$	Colored task; $\tilde{\mathcal{B}} = \Sigma_x^{1/2}\mathcal{B}$
$\tilde{\kappa}_\ell$	Cumulants of $\tilde{\mathcal{B}}$
	Kernel and eigenstructure
c_ℓ	Level coefficients of an inner-product kernel (Definition 1)
K_ℓ	Degree-ℓ subkernel (Definition 1)
$\lambda_i,\ \phi_i$	Kernel eigenvalues and (orthonormal) eigenfunctions
v_i	Eigencoefficient, $f^\star = \sum_i v_i\phi_i$
$\mathcal{L}_i$	Learnability of mode i; $\hat{f} = \sum_i \mathcal{L}_i v_i \phi_i$
ζ	Eigenframework constant solving Eq. (18)
$\mathcal{E}_0$	Overfitting coefficient
$C_{FF}, C_{FL}, C_{LL}, \tilde{C}_\ell$	Level covariance blocks and projected covariance of Definition 4
$S_{ij},\ \tilde{v}_i$	Eigenfunction overlap and test eigencoefficients on μ_{OOD}

Table 1: Quantities appearing throughout the main document and appendices.

B Experimental Details

In Figure 1, we introduced g_i to describe polynomial functions learned by the kernel. The functions g_i are chosen to be representative of the polynomials of degree $2i+1$ with large eigencoefficients: that is, g_0 was randomly selected as one of the d eigenfunctions $\boldsymbol{x}_{L+1}$, g_1 was chosen to be eigenfunction most closely related to $\phi_3(\boldsymbol{z}) \sim \left(\sum_i y_i \boldsymbol{x}_i^\top - \frac{L}{d}\hat{\boldsymbol{\mu}}_{\boldsymbol{\beta}}^\top\right)\boldsymbol{x}_{L+1}$, g_2 was one of the d eigenfunctions most closely related to (at leading order) $\phi_5 \sim \sum_{ij} y_i y_j \boldsymbol{x}_i^\top \boldsymbol{x}_j \boldsymbol{x}_{L+1}$, and finally g_3 was the eigenfunction most closely related to (at leading order) $\phi_7 \sim \sum_{ijk} y_i y_j y_k \boldsymbol{x}_i^\top \boldsymbol{x}_j \boldsymbol{x}_k^\top \boldsymbol{x}_{L+1}$.

B.1 Exact experimental setups

We ran three main types of experiments: exact kernel experiments, exact random feature solvers, and random feature (RF) model solutions found via SGD. In the main text, we detail only the kernel experiments and RF model experiments, as those were the most fruitful. Below, we describe the setup for each figure in the main text, along with our rationale for using random feature models in place of kernels.

The ICL Kernel. A kernel method solving the ICL linear regression task requires an enormous number of samples to get to either of the Bayes-optimal predictors. We can, however, reduce this by very carefully choosing a kernel function constructed to reduce as many terms with eigencoefficient 0 as possible. We use the subkernel expansion framework (Definition 4) and the eigencoefficients found in Section F, to make the "ICL Kernel", a Gaussian kernel with the linear components of $\boldsymbol{x}_{L+1}$ extracted, and the even polynomials removed:

$$K_{ICL}(\boldsymbol{z}, \boldsymbol{z}') = \Big(\exp\big(-|\boldsymbol{c} - \boldsymbol{c}'|^2/2\sigma^2\big) + \exp\big(-|\boldsymbol{c} + \boldsymbol{c}'|^2/2\sigma^2\big) \Big) \boldsymbol{x}_{L+1}^\top \boldsymbol{x}'_{L+1}, \tag{6}$$

where $\boldsymbol{c}$ is nothing but $\boldsymbol{z}$ with the unpaired datapoint $\boldsymbol{x}_{L+1}$ removed.

Kernel model experiments. In the right half of Figure 1, we train an ICL kernel with kernel width $\sigma = 3$, a ridge parameter $\delta = 10^{-2}$, and each individual sample partially whitened such that $\sum_i \boldsymbol{x}_i \boldsymbol{x}_i^\top = I_d/d$. We take $d = 5$, $L = 7$, and $\epsilon^2 = 0.1$. The in-context data is sampled $\boldsymbol{x}_i \sim \mathcal{N}(0, I_d/(dL)$. We draw $T = 16$ tasks to get $\mathcal{B}_{\text{tr}}$, taken from a prior of $\mathcal{B}_{\text{prior}} \sim \mathcal{N}(0, I_d)$. For the trained kernel models, we use 20 trials, keeping $\mathcal{B}_{\text{tr}}$ fixed.

Direct eigenstructure checks. In Figure 7, we check that the subkernel expansion framework (Definition 4) accurately predicts the eigendecomposition of a trained kernel. The two spectra are straightforward to compare: we sort both in decreasing order and overlay them against mode index on log–log axes. Agreement means the curves coincide. The eigenbases are more difficult, most notably due to the ordering. The spectrum is degenerate over many different sections, and so assigning a cosine similarity between a theoretical eigenvector and empirical one mode-by-mode is at best left to chance. Therefore, we must bin the eigenvectors and compare their overall overlap.

We instead place bin edges in the gaps of the spectrum between degenerate sectors at the geometric mean $\sqrt{\lambda_{(m)}\lambda_{(m+1)}}$ of each consecutive pair whose ratio exceeds a hyperparameter threshold r. We additionally consolidate many bins at smaller eigenvalues (defined by another hyperparameter) into 2 bins such that our images are not dominated by many bins that are empirically noisy.

Let $d_\ell^{(\text{emp})}$ be the number of empirical modes in bin ℓ. Let $\Phi_\ell^{(\text{emp})} \in \mathbb{R}^{n \times d_\ell^{(\text{emp})}}$ collect the empirical eigenvectors of bin ℓ as columns, and let Φ_k do the same for the predicted eigenfunctions of bin k. The pairwise subspace overlap is

$$\mathrm{O}(k, l) = \frac{1}{d_l^{(\text{emp})}} \left\| \Phi_k^\top \Phi_l^{(\text{emp})} \right\|_F^2. \tag{7}$$

The bottom plots ofFigure 7 shows $\mathrm{O}(k, l)$ as a heatmap with eigenvalue increasing towards the top left, so that the horizontal axis runs in the same direction as mode index in the panel above.

We use $n = 25600$ samples. Our setup consistently uses $d = 4$, $\epsilon^2 = 0.1$, $\Sigma_{\boldsymbol{x}} = I_d/(dL)$, and $\Sigma_{\boldsymbol{\beta}} = I_d$. All setups use $L = 6$, outside of the Laplace kernel which uses $L = 9$ such that the the inner-product kernel expansion remains valid (concentration of norm of $\|Z\|$), as is described in more detail in B.3 of (Karkada et al., 2026). Outside of the Laplace kernel for norm concentration reasons, all kernel:task-diversity pairs were chosen at random. In order of their appearance:

- T=1, Inner-product kernel uses level coefficients $c_0 = 1$, $c_1 = 1/9$, $c_2 = 1/81$, $c_3 = 1/729$, with bid edges of $r = 1.01$, and the coarse graining of the tail beginning at $\lambda = 10^{-5}$, ending at $\lambda = 10^{-7}$.
- T=4, ReLU NTK uses bias and weight variances of 1, bin edges of $r = 1.018$, and the coarse graining of the tail beginning at $\lambda = 7 \cdot 10^{-5}$, ending at $\lambda = 6 \cdot 10^{-6}$.
- T=16, Gaussian kernel uses kernel width of 3, with bid edges of $r = 1.008$, and the coarse graining of the tail beginning at $\lambda = 10^{-4}$, ending at $\lambda = 10^{-6}$.
- T=256, Laplace kernel uses kernel width of 3, with bid edges of $r = 1.01$, and the coarse graining of the tail beginning at $\lambda = 8 \cdot 10^{-5}$, ending at $\lambda = 10^{-5}$.

Learning curves. In **Figure 8**, we train a Gaussian kernel with kernel width parameter $\sigma^2 = 3 \cdot 7 = 21$ and ridge $\delta = 0.1$, on isotropic ICL data with $d = 5$ intrinsic dimensions, context length $L = 7$, and noise variance $\epsilon^2 = 0.1$. A large kernel width is used as our raw contextual data scales linearly with the context length L: $\|Z\| \sim 2L + 1$, $2L$ from $\|\boldsymbol{x}\| \sim O(1)$ and $\|y_i\| \sim O(1)$. Both

in-context data $\boldsymbol{x}$ and the tasks $\mathcal{B}$ are drawn isotropically as $\boldsymbol{x} \sim \mathcal{N}(0, I_d/d)$ and $\boldsymbol{\beta} \sim \mathcal{N}(0, I_d)$. All runs are performed across 20 seeds with the max n_{samples} set by the memory constraints of a Google Colab A-100 GPU: roughly 25,600 samples. We sweep over 4 logspaced values of task diversity T: $1, 4, 16, 256$ to show our theoretical eigensystem is correct regardless of task diversity and regardless of what distribution we evaluate our theory on.

B.2 Random Feature Analytical Solver

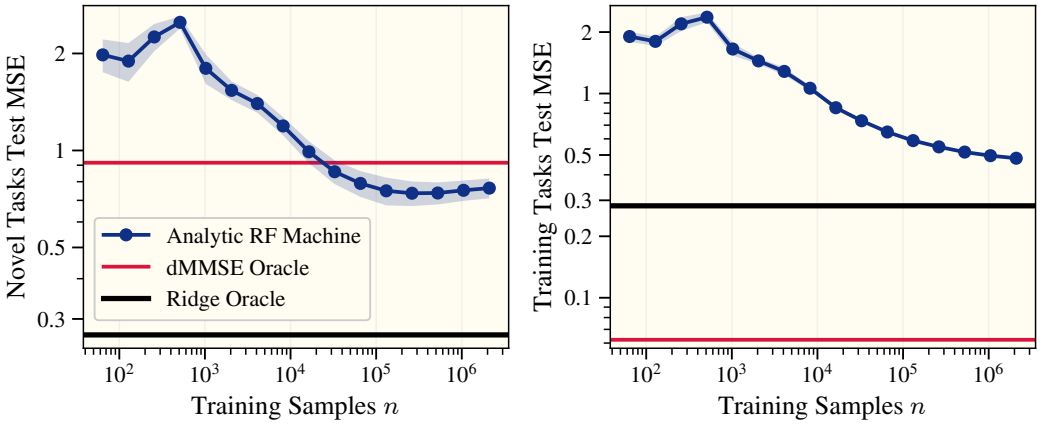


Figure 4: Analytically solved random feature models outperform the memorizing oracle on novel tasks. We show test MSE on novel tasks ($\boldsymbol{\beta} \sim \mathcal{B}_{\text{Prior}}$) and on the training tasks $\boldsymbol{\beta} \in \mathcal{B}_{\text{tr}}$ of a random feature model whos readout layer weight has been analytically solved, as a function of the number of training samples n, The limiting kernel function is the ICL kernel of Equation (6), set with kernel width $\sigma = 3$. We use $T = 8$, $L = 4$, $d = 4$, $\epsilon^2 = 0.05$, and tasks drawn from an isotropic prior. Shading shows the standard deviation over 5 trials. Past $n \approx 4 \times 10^3$, the model's error drops below that of the memorizing oracle (dMMSE). It remains well above the generalizing oracle (ridge) throughout.

As exact kernel solvers scaled poorly with the number of samples (we need to inverse an $n \times n$ matrix), we looked towards using random feature models instead. Kernel models with suitable kernels can be understood as an infinte-width two-layer MLP with the first layer frozen (i.e. a fixed feature map followed by a linear projection) (Rahimi & Recht, 2007). Random feature (RF) models serve as their finite-width counterpart, so long as the random features defining the frozen layer give the correct NNGP in the limit that the number of features approaches ∞. RF models have a computational advantage: with p random features, fitting the readout by ridge regression only requires solving a $p \times p$ linear system, reducing the training cost from $\mathcal{O}(n^3)$ time and $\mathcal{O}(n^2)$ memory to $\mathcal{O}(np^2 + p^3)$ time and $\mathcal{O}(np)$ memory.

So long as we are not limited by the number of features (a feature-limited regime), we can gain extra performance from the RF models. To see we're in a *data-limited* regime, we can note that we're mostly noise-dominated, as the non-zero eigencoefficients are sparsly populating our spectrum, and worse than that, the eigencoefficients aren't particularly contained within the large eigenvalue region (the largest eigencoefficients appear at the degree-3 part of the spectrum corresponding to $\sum_i y_i \boldsymbol{x}_i^\top \boldsymbol{x}_{L+1}$, for which we have to fit the noisy degree 1 and 2 eigenfunctions. Therefore, we attempted to use a random feature model to substantiate our claims, directly solving for the optimal readout layer. This culminates in Figure 4, where we can see that it's somewhat difficult to see our theory's predicted characteristics (though the drop and rise still are present).

B.3 Random Feature Experiments

While still facing memory issues, we moved to non-analytic solvers for the fixed-feature models. As previously mentioned, the RF model is nothing but a special type of two-layer MLP, and training said MLP with online SGD would allow us to trade memory constraints for runtime.

In **Figure 3**, we train an RF model using SGD using $3.3 \cdot 10^6$ gradient steps with a batch size of 256. The dataset uses $d = 4, L = 5, \epsilon^2 = 0.05$. Our RF models use random features drawn such that they converge to the ICL kernel of Equation (6). The kernel width parameter used for the converging kernel was chosen as $\sigma = 1.1\|\boldsymbol{z}\|$. We draw $T = 8$ tasks $\mathcal{N}(0, \boldsymbol{I}_d)$. Our data is drawn $\boldsymbol{x} \sim \mathcal{N}(0, I_d/(d \cdot 4))$,

the numerical factor of 4 added such that $d\sigma_x^2 = \epsilon^2$ (which we found best portrayed our theory). We normalize our labels y by $d\sigma_x^2 + \epsilon^2 = 1/2$ to keep the scale invariant to choice of Σ_x. We run our experiments and average across 3 trials, shading the standard deviation. We used only 8192 features for our model. We use a learning rate $\eta = 0.75 \cdot \frac{2}{\lambda_1}$, λ_1 being a numerically estimated maximum eigenvalue of the Hessian found via power iteration. To reduce random noise, we weight average the random feature model weights of at most $20,000$ steps, and exit our training loop prematurely once we are within a tolerance of 0.01 of the predicted optimal polynomial degree 3 solution.

In Figure 5, we train a RF model on $d = 3$ dimensional data with *only* $T = 4$ tasks. We use the same ICL kernel of Equation (6) as our limiting kernel for which the features are drawn. The kernel width parameter used for the converging kernel was chosen as $\sigma = 1.5\|\boldsymbol{z}\|$. These tasks are initially drawn from a white Gaussian prior, but are manually whitened and centered: $\mathrm{Cov}_{\mathcal{B}_{\mathrm{tr}}} = \boldsymbol{I}_d$, and the mean training task is set to $\mathbf{0}$. We take a large context length of $L = 96$ and $\epsilon^2 = 0.05$. We draw data $\boldsymbol{x} \sim \mathcal{N}(0, I_d/d)$. We perform only 1 trial. We normalize our labels y by $d\sigma_x^2 + \epsilon^2 = 1.05$ to keep the scale invariant to choice of Σ_x. We use $65,536$ features for our model. We train for $3.2 \cdot 10^6$ gradient steps with a batch size of 256. We use a learning rate $\eta = 0.8 \cdot \frac{2}{\lambda_1}$, λ_1 being a numerically estimated maximum eigenvalue of the Hessian found via power iteration.

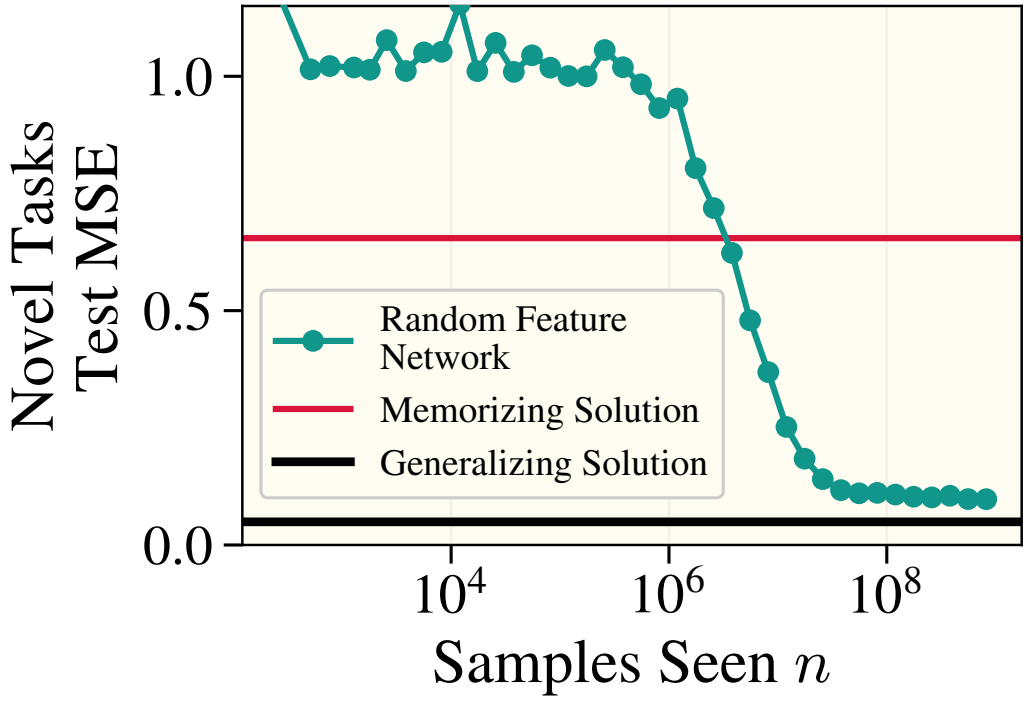


Figure 5: Random feature models can perform ICL even with only $T = d + 1$ tasks. We train a random feature network using SGD with exactly $T = 4 = d + 1$ tasks ($d = 3$), with context length $L = 96$. We evaluate said model's test error on novel tasks $\beta^\star \sim \mathcal{N}(0, I)$, normalized by $\mathbb{E}[y_q^2]$. We manually construct a training task pool with $\hat{\Sigma} = I$, $\hat{\mu} = 0$ and $\mathrm{Tr}_1 \hat{\kappa}_3 = 0$. Under these conditions, the optimal polynomial degree 3 truncation of the memorizing solution is, up to finite context length L, the generalizing solution exactly. Even though the model will eventually memorize the task given infinite training time and random features, it approaches the optimal degree 3 truncation of f_{mem}, getting error equivalent to that of a generalizing solution. Further details in Section B.

C Cumulants Review

Our results are phrased in terms of the cumulants of the task pool. Here, we review cumulants and the few facts the main text uses.

Definition. Write $\mathbb{E}_{\mathcal{B}_{\rm tr}}[g(\boldsymbol{\beta})] = \frac{1}{T}\sum_{t=1}^{T} g(\boldsymbol{\beta}^{(t)})$ for the average over the uniform measure on the pool. Its cumulant generating function is

$$\hat{K}(\mathbf{t}) = \log \mathbb{E}_{\mathcal{B}_{\rm tr}}\left[e^{\mathbf{t}^\top \boldsymbol{\beta}}\right], \tag{8}$$

and the order-ℓ cumulant $\hat{\kappa}_\ell$ is the symmetric tensor of its ℓ-th derivatives at the origin,

$$(\hat{\kappa}_\ell)_{a_1\cdots a_\ell} = \left.\frac{\partial^\ell \hat{K}}{\partial t_{a_1}\cdots\partial t_{a_\ell}}\right|_{\mathbf{t}=0}. \tag{9}$$

We write $M_\ell = \mathbb{E}_{\mathcal{B}_{\rm tr}}[\boldsymbol{\beta}^{\otimes \ell}]$ for the raw moment tensors.

Relation to moments. With $\delta\boldsymbol{\beta} = \boldsymbol{\beta} - \hat{\mu}_{\boldsymbol{\beta}}$ the centered task, the first four cumulants are

$$\hat{\kappa}_1 = \hat{\mu}_{\boldsymbol{\beta}}, \tag{10}$$

$$\hat{\kappa}_2 = \mathbb{E}_{\mathcal{B}_{\rm tr}}\left[\delta\boldsymbol{\beta}\,\delta\boldsymbol{\beta}^\top\right] = \hat{\Sigma}_{\boldsymbol{\beta}}, \tag{11}$$

$$(\hat{\kappa}_3)_{abc} = \mathbb{E}_{\mathcal{B}_{\rm tr}}[\delta\beta_a\,\delta\beta_b\,\delta\beta_c], \tag{12}$$

$$(\hat{\kappa}_4)_{abcd} = \mathbb{E}_{\mathcal{B}_{\rm tr}}[\delta\beta_a\,\delta\beta_b\,\delta\beta_c\,\delta\beta_d] - \left(\hat{\Sigma}_{ab}\hat{\Sigma}_{cd} + \hat{\Sigma}_{ac}\hat{\Sigma}_{bd} + \hat{\Sigma}_{ad}\hat{\Sigma}_{bc}\right). \tag{13}$$

In general, $\hat{\kappa}_\ell$ is the order-ℓ central moment with all relevant lower-order information subtracted—it is the information in the order-ℓ moment not already carried by lower orders.

Key properties used throughout our findings. Cumulants beyond the first are invariant under shifts, and all cumulants are multilinear under linear maps: if $\tilde{\boldsymbol{\beta}} = A\boldsymbol{\beta}$, then

$$(\tilde{\kappa}_\ell)_{a_1\cdots a_\ell} = \sum_{b_1,\ldots,b_\ell} A_{a_1 b_1}\cdots A_{a_\ell b_\ell}\,(\hat{\kappa}_\ell)_{b_1\cdots b_\ell}. \tag{14}$$

Taking $A = \Sigma_{\boldsymbol{x}}^{1/2}$ gives the colored cumulants $\tilde{\kappa}_\ell$ of Section F.

A Gaussian has all its cumulants beyond order two vanish. The prior $\mathcal{N}(\mu_{\boldsymbol{\beta}}, \Sigma_{\boldsymbol{\beta}})$ therefore carries only a mean and a covariance, but a *finite* pool of T draws from it does not. Each $\hat{\kappa}_\ell$ is a sample statistic that converges to the prior's value as $T \to \infty$ with entrywise fluctuations of order $T^{-1/2}$, so $\hat{\kappa}_1 \to \mu_{\boldsymbol{\beta}}$, $\hat{\kappa}_2 \to \Sigma_{\boldsymbol{\beta}}$, and $\hat{\kappa}_{\ell\geq 3} \to 0$. This rate is entrywise at fixed d: tensor norms carry additional factors of d, and when d is comparable to T the spectrum of $\hat{\Sigma}_{\boldsymbol{\beta}}$ follows the Marchenko–Pastur law rather than concentrating at the prior's eigenvalues (Section 7).

Several eigencoefficients depend on cumulants only through partial traces. We write Tr_k for the contraction of k index pairs,

$$(\mathrm{Tr}_k\,\hat{\kappa}_\ell)_{a_{2k+1}\cdots a_\ell} = \sum_{b_1,\ldots,b_k} (\hat{\kappa}_\ell)_{b_1 b_1\cdots b_k b_k\, a_{2k+1}\cdots a_\ell}, \tag{15}$$

which by symmetry of $\hat{\kappa}_\ell$ does not depend on which pairs are contracted. For example, $\mathrm{Tr}_1\hat{\kappa}_3$, with components $\sum_a(\hat{\kappa}_3)_{aab}$, enters the degree-3 coefficients, and $\mathrm{Tr}_2\hat{\kappa}_5$, with components $\sum_{a,b}(\hat{\kappa}_5)_{aabbc}$, enters the degree-5 coefficients. We call a pool *traceless* at order ℓ if $\mathrm{Tr}_1\hat{\kappa}_\ell = 0$. Contracted cumulants can therefore appear at lower polynomial degree than the full tensor, which first appears at degree $2\omega - 1$ for $\hat{\kappa}_\omega$ (Theorem 1).

D Review of the KRR Eigenframework for Predicting Test Performance from Task Eigenstructure

The central piece of existing theory which we use in this paper is a set of equations which give the average-case test MSE of KRR in terms of the task eigenstructure. In this appendix, we will review this KRR eigenframework.

This eigenframework has been derived many times by different means in both the statistical physics community and the classical statistics community (which usually phrases the result as applying to linear ridge regression). References studying KRR include Sollich (2001); Bordelon et al. (2020); Jacot et al. (2020); Simon et al. (2021); Loureiro et al. (2021); Wei et al. (2022a); references studying linear ridge regression include Caponnetto & De Vito (2007); Dobriban & Wager (2018); Wu &

Xu (2020); Hastie et al. (2022); Richards et al. (2021); Cheng & Montanari (2022). The result is essentially the same in all cases. Here we will adopt the terminology and notation of Simon et al. (2021).

Recall that we are studying KRR with a kernel function K with data sampled $\boldsymbol{x}_i \sim \mu$, targets generated as $y_i = f_*(\boldsymbol{x}_i) + \eta$, and noise $\eta \sim \mathcal{N}(0, \epsilon^2)$ with variance $\epsilon^2 \geq 0$. The kernel admits an eigendecomposition $K(\boldsymbol{x}, \boldsymbol{x}') = \sum_i \lambda_i \phi_i(\boldsymbol{x}) \phi_i(\boldsymbol{x}')$ with orthonormal eigenfunctions (ϕ_i). Let us decompose the target function in the eigenbasis as $f_*(\boldsymbol{x}) = \sum_i v_i \phi_i(\boldsymbol{x})$. Suppose we run KRR with n samples and a ridge parameter $\delta \geq 0$, and we compute the population (i.e. test) and train MSEs as

$$\texttt{MSE}_{\text{te, ID}} = \mathop{\mathbb{E}}_{\boldsymbol{x} \sim \mathcal{D}} \left[(f_*(\boldsymbol{x}) - \hat{f}(\boldsymbol{x}))^2 \right] + \epsilon^2, \tag{16}$$

$$\texttt{MSE}_{\text{tr}} = \frac{1}{n} \sum_i (y_i - \hat{f}(\boldsymbol{x}_i))^2. \tag{17}$$

D.1 Statement of the eigenframework

We are now ready to state the eigenframework. Let $\zeta \geq 0$ be the unique nonnegative solution to

$$\sum_i \frac{\lambda_i}{\lambda_i + \zeta} + \frac{\delta}{\zeta} = n. \tag{18}$$

Then test risk is given approximately by

$$\texttt{MSE}_{\text{te}} \approx \mathcal{E}_{\text{te}} := \mathcal{E}_0 \text{B}, \tag{19}$$

where the *overfitting coefficient* $\mathcal{E}_0$ is given by

$$\mathcal{E}_0 := \frac{n}{n - \sum_i \left(\frac{\lambda_i}{\lambda_i + \zeta} \right)^2} \tag{20}$$

and the *bias* is given by

$$\text{B} = \sum_i \left(\frac{\zeta}{\lambda_i + \zeta} \right)^2 v_i^2 + \epsilon^2. \tag{21}$$

Train risk is given by

$$\texttt{MSE}_{\text{tr}} \approx \mathcal{E}_{\text{tr}} \equiv \frac{\delta^2}{n^2 \zeta^2} \mathcal{E}_{\text{te}}, \tag{22}$$

What is meant by the "$\approx$" in Equations (19) and (22)? This result only becomes exact in certain stringent cases; it is formally derived under an assumption that the eigenfunctions are independent Gaussian (or sub-Gaussian) variables when $\boldsymbol{x}$ is sampled from μ, and it is exact only in an asymptotic limit in which n and the number of eigenmodes in a given eigenvalue range (or the number of duplicate copies of any given eigenmode) both grow large at a proportional rate (Hastie et al., 2022; Bach, 2023). These conditions emphatically do not apply to any realistic instance of KRR. Nonetheless, numerical experiments find that Equations (19) and (22) hold with small error even at modest n (Canatar et al., 2021; Simon et al., 2021; Wei et al., 2022a): though derived in an idealized setting and exact only in a limit, this eigenframework holds reliably in practical cases.

In this paper, we use this eigenframework as a tool to map predictions of task eigenstructure to predictions of learning curves. Since we are here using it in settings very similar to those tested by previous works (Bordelon et al., 2020; Jacot et al., 2020; Simon et al., 2021; Wei et al., 2022a), we expect it to work well. Its use introduces some small error (as it is not perfect at finite n), but this is usually not the dominant source of error.

We have found numerically that this framework fails on ICL problems with a single intrinsic dimension, $d = 1$, but works well otherwise. This failure mode is due to the Gaussian universality condition breaking; see Figure 6.

D.2 The Out-of-Distribution Eigenframework

While the eigenframework as presented in Sollich (2001); Bordelon et al. (2020); Jacot et al. (2020); Simon et al. (2021); Loureiro et al. (2021); Wei et al. (2022a) specify only the learning dynamics when

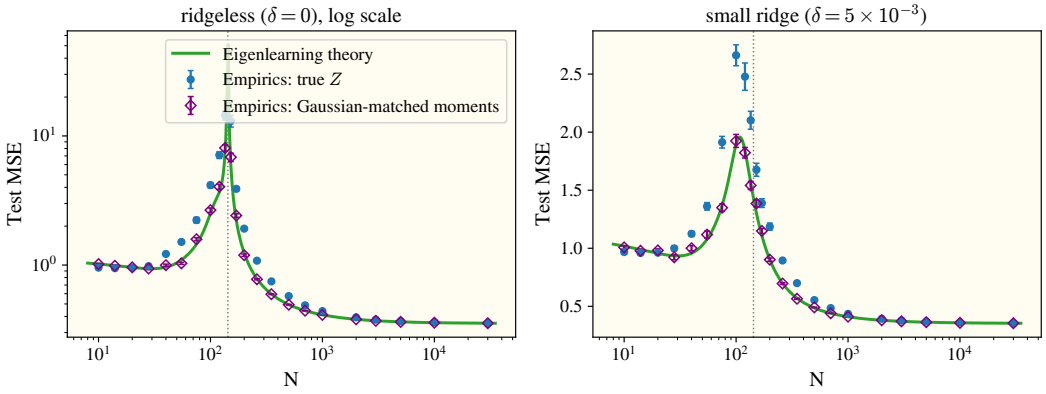


Figure 6: One-dimensional ICL data breaks Gaussian universality. We compare the eigenframework's predicted test MSE (line) against kernel regression trained on true ICL contexts (circles) and on a Gaussian surrogate with matched moments (diamonds), for $T = 4$, $d = 1$, $L = 8$. We use a Gaussian kernel with kernel width $\sigma^2_{\text{ker}} = 3$ and in-context data with covariance $\Sigma_{\boldsymbol{x}} \sim I_d/(dL)$. The left panel is ridgeless ($\delta = 0$) and the right uses a small ridge ($\delta = 5 \times 10^{-3}$). The Gaussian surrogate tracks the theory at every n. The true contexts agree away from the interpolation peak but exceed the prediction near it, most visibly at small ridge. Both datasets share the same theoretical curve, so the mismatch comes from the non-Gaussianity of one-dimensional ICL data rather than from the predicted eigenstructure. We observe no such failure for $d > 1$.

the distribution of the trainset is equivalent to that of the testset, our techniques require specifying the test error on a novel "ICL" distribution. Luckily, an out-of-distribution version of the eigenframework appears in Canatar et al. (2022), which we can use as a baseline, choosing to adapt the same notation as Simon et al. (2021). The main difference between the in-distribution and an out-of-distribution test set is that our eigenfunctions need no longer be *eigen* (orthonormal) on our new distribution. Therefore, the bulk of our work will be in describing this non-orthonormality. We begin by defining two useful quantities going forward, the eigenfunction overlap and the test eigencoefficients:

$$S_{ij} = \langle \phi_i(Z)\phi_j(Z) \rangle_{\mathcal{D}_{\text{OOD}}} \tag{23}$$

$$\tilde{v}_i = S^{-1}_{ij} \langle y^\star \phi_j(Z) \rangle_{\mathcal{D}_{\text{OOD}}} \tag{24}$$

Writing $y^\star$ as a sum of a resolvable and unresolvable portion,

$$y^\star(Z) = \sum_i \tilde{v}_i \phi_i(Z) + f_\perp, \tag{25}$$

we can expand $\texttt{MSE}_{\text{te, OOD}}$ as,

$$\texttt{MSE}_{\text{te, OOD}} = \mathop{\mathbb{E}}_{\boldsymbol{x} \sim \mathcal{D}_{\text{OOD}}} \left[(f_*(\boldsymbol{x}) - \hat{f}(\boldsymbol{x}))^2 \right] + \epsilon^2 \tag{26}$$

$$= \mathop{\mathbb{E}}_{\boldsymbol{x} \sim \mathcal{D}_{\text{OOD}}} \left[(\hat{v} - \tilde{v})^\top S (\hat{v} - \tilde{v}) \right] + ||f_\perp||^2_{\text{OOD}} + \epsilon^2 \tag{27}$$

$$= (\mathbb{E}[\hat{v} - \tilde{v}])^\top S (\mathbb{E}[\hat{v} - \tilde{v}]) + \text{tr}[S \cdot Cov(\hat{v})] + ||f_\perp||^2_{\text{OOD}} + \epsilon^2 \tag{28}$$

$$= \Delta^\top S \Delta + \frac{\mathcal{E}_0}{n} \sum_i S_{ii} \mathcal{L}_i^2 \left(\sum_j (1 - \mathcal{L}_j)^2 v_j^2 + \epsilon^2 \right) + ||f_\perp||^2_{\text{OOD}} + \epsilon^2 \tag{29}$$

where $\Delta = \tilde{v} - \mathcal{L}v$. The first part of the equation ($\Delta^\top S \Delta$) can be understood to be the "out-of-distribution bias" term, the second part as the "out-of-distribution variance" term, and the last ($||f_\perp||^2_{\text{OOD}} + \epsilon^2$) as an unresolvable noise.

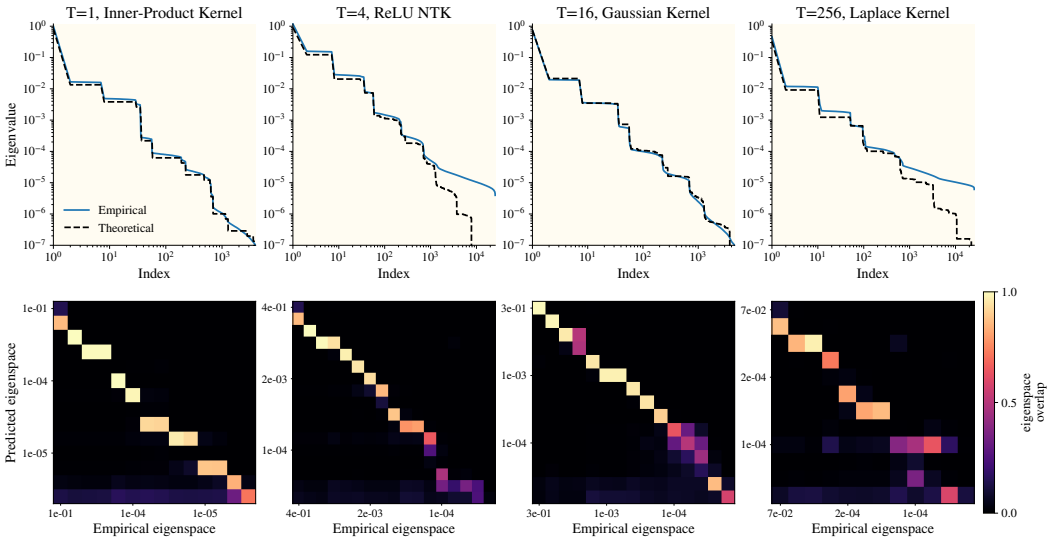


Figure 7: The subkernel expansion framework (4) accurately predicts the eigenstructure of in-context learning data for various rotation-invariant kernels. Across various rotation-invariant kernels with varying task diversity (columns), we compute the empirical kernel eigensystem $\{(\lambda_i^{(\text{emp})}, \phi_i^{(\text{emp})})\}$ and compare to the theoretical eigensystem $\{(\lambda_i^{(\text{th})}, \phi_i^{(\text{th})})\}$ obtained from Definition 4, indexed in order of decreasing $\lambda_i^{(\text{th})}$. In the top row, we overlay the eigenvalue spectrum between $\lambda_i^{(\text{emp})}$ and $\lambda_i^{(\text{th})}$. In the bottom row, we bin both the predicted and empirical eigenfunctions into shared spectral bins and visualize the pairwise subspace overlap ((7)). Concentration along the diagonal indicates theory-experiment match. See Section B for further explanation and experimental details.

E Details on the Kernel Eigenstructure

The kernel's prediction is governed by its (eigenvalue, eigenfunction) pairs (λ_i, ϕ_i) with respect to the training distribution $\mathcal{D}_{\text{tr}}$,

$$\mathbb{E}_{\boldsymbol{z}' \sim \mathcal{D}}\left[K(\boldsymbol{z}, \boldsymbol{z}')\phi_i(\boldsymbol{z}')\right] = \lambda_i \phi_i(\boldsymbol{z}), \qquad \mathbb{E}_{\boldsymbol{z} \sim \mathcal{D}}\left[\phi_i \phi_j\right] = \delta_{ij}. \tag{30}$$

This is analytically intractable for the ICL data and kernels that we analyze, so we need instead to find the eigendecomposition another way. Here, we walk through how to arrive at the subkernel expansion framework, our method of approximating the eigensytem, then give its full definition. The ICL regression task does not have closed-form eigenfunctions in full generality, but we can nonetheless approximate them well.

E.1 The subkernel expansion framework

We first give a brief primer on what the subkernel expansion framework is at a high level before giving the technical definition, much in the same way integrals are first introduced using Riemann sums.

Because $\|\boldsymbol{z}\|$ concentrates for ICL contexts (Section E), the rotation-invariant kernels are well approximated by inner-product kernels (Definition 1), which depend on their inputs only through $\boldsymbol{z}^\top \boldsymbol{z}'$.

Definition 1 (Inner-product kernel and its subkernels)**.** A kernel is an *inner-product kernel* if $K(\boldsymbol{z}, \boldsymbol{z}') = \sum_{\ell \geq 0} \frac{c_\ell}{\ell!} (\boldsymbol{z}^\top \boldsymbol{z}')^\ell$ with "level coefficients" $c_\ell \geq 0$, which guarantees K is PSD. Its degree-ℓ *subkernel* is the single term $K_\ell(\boldsymbol{z}, \boldsymbol{z}') = \frac{c_\ell}{\ell!} (\boldsymbol{z}^\top \boldsymbol{z}')^\ell$.

We begin with an intuitive explanation for the eigenfunctions. We treat each subkernel of Definition 1 separately. The degree-1 subkernel is nothing but a linear kernel (*i.e.* performing linear regression), so its eigenfunctions are the principal components of the raw data. Since $(\boldsymbol{z}^\top \boldsymbol{z}')^\ell = (\boldsymbol{z})_\ell^\top (\boldsymbol{z}')_\ell$, where $(\boldsymbol{z})_\ell$ collects all degree-ℓ polynomial of $\boldsymbol{z}$, the degree-ℓ subkernel is likewise a linear kernel on the degree-ℓ polynomial of the data, and we might expect its eigenfunctions to be their principal

components. This fails because polynomial overlap *across* degrees (*e.g.* x^2 with the constant). To handle this, we impose the assumption that level coefficients decay quickly, $c_{\ell+1}\,\mathbb{E}\|\boldsymbol{z}\|^2 \ll c_\ell$. Each shared component is then claimed by the lowest degree that contains it. The degree-ℓ eigenfunctions are therefore the principal components of the degree-ℓ polynomial after all lower degrees are projected out, with eigenvalues as $c_\ell/\ell!$ times the variances of those components.

The terms of the subkernel expansion with ICL data In the walkthrough, we labeled the terms simply as "polynomials," although it's helpful to discuss which polynomials we are talking about, and which will be used for a model's computation. We find that only a select subset will have non-0 coefficient when expanding functions in the kernel's eigenbasis, which we label as subkernel expansion ICL polynomials. These serve as good approximations to the eigenfunctions, with the upside that they're still interpretable.

Definition 2 (Subkernel expansion ICL polynomials)**.** The degree p Subkernel expansion ICL polynomials, $\Psi_p(Z)$, are a set of functions ψ_p that can be written as a tensor product of p components of the context Z in the following manner:

$$\psi_p(Z) \equiv \left(\bigotimes_{i=1}^{m}\sum_{\ell=1}^{L} y_\ell^{a_i}\boldsymbol{x}_\ell^{\otimes b_i}\right)\otimes \boldsymbol{x}_{L+1} \tag{31}$$

where $p = 1 + \sum_{i=1}^{m}(a_i + b_i)$. Ψ_p contains multiple different polynomial functions, which we enumerate with $\psi_p, \psi_p', \psi_p^{(2)}, \ldots, \psi_p^{(k)}$. For instance, the degree 3 Subkernel expansion ICL polynomials contain both the functions $\psi = \sum_i y_i \boldsymbol{x}_i \otimes \boldsymbol{x}_{L+1}$ of size d^2 and $\psi_3' = \sum_i y_i^2 \boldsymbol{x}_{L+1}$ of size d.

Although it would be mathematically simple to work with Subkernel expansion ICL polynomials, we cannot, as they couple across different orders. For instance, Ψ_5 overlaps nontrivially with Ψ_3, e.g. $\psi_5 = \sum_i y_i^3 \boldsymbol{x}_i \boldsymbol{x}_{L+1}$ and $\psi_3 = \sum_i y_i \boldsymbol{x}_i \boldsymbol{x}_{L+1}$ are not orthogonal as $\mathbb{E}_{\mathcal{D}}[y_i^2] > 0$. To account for this overlap, we invoke that our kernels will learn the optimal low-degree function first (*i.e.* the lower-degree functions will have larger eigenvalues), and therefore project out the lower-degree information from the higher degree polynomial. We label these processed terms as the "Subkernel expansion ICL perpendicular polynomials."

Definition 3 (Subkernel expansion ICL perpendicular polynomials)**.** A degree p set of subkernel expansion ICL perpendicular polynomials, $\Psi_p^{\perp}(Z)$, is constructed as a corresponding subkernel expansion ICL polynomial (Definition 2) with all overlap between lower-degree subkernel expansion ICL polynomials projected out:

$$\Psi_p^{\perp}(Z) \equiv \Psi_p(Z) - \mathbb{E}_{\mathcal{D}}\left[\Psi_p \Psi_{<p}^{\top}\right]\mathbb{E}_{\mathcal{D}}\left[\Psi_{<p}\Psi_{<p}^{\top}\right]^{+}\Psi_{<p}(Z) \tag{32}$$

where each function of Ψ_p is subtracted against *every* subkernel expansion ICL polynomial of a lower degree, and $(\cdot)^+$ denotes a pseudoinverse. To note, we do not make any subtractions within a degree.

The subkernel expansion ICL perpendicular polynomials are now orthogonal across degrees, but not within a given degree: the polynomials share within-context fluctuations and become correlated through the task distribution's statistics. For example, the overlap between $\psi_3^{\perp} = \sum_i y_i \boldsymbol{x}_{i,1}\boldsymbol{x}_{L+1,1} \approx \sum_i \boldsymbol{\beta}_1^{\top} I_d \boldsymbol{x}_{L+1,1}/d$ and $\psi_3^{\perp} = \sum_i y_i \boldsymbol{x}_{i,2}\boldsymbol{x}_{L+1,1} \approx \sum_i \boldsymbol{\beta}_2^{\top} I_d \boldsymbol{x}_{L+1,1}/d$ is non-zero unless the covariance of the training tasks is isotropic ($\hat{\Sigma}_{\boldsymbol{\beta}} \propto I_d$). To account for this, we orthonormalize each degree as a single block using its Gram matrix to obtain the kernel's predominately useful eigenfunctions, the ones with non-0 eigencoefficient.

The subkernel expansion framework, in full detail Here, we provide the subkernel expansion framework.

Definition 4 (Subkernel expansion)**.** The subkernel expansion of a kernel K is obtained using the following procedure:

1. Expand the chosen kernel function as an inner-product kernel (Definition 1) with level coefficients c_ℓ.
2. Let $(\boldsymbol{Z})_\ell$ be the vector of all entries of $\boldsymbol{z}^{\otimes \ell}$, i.e. all ordered products of ℓ coordinates of $\boldsymbol{z}$, so that $(\boldsymbol{z}^\top \boldsymbol{z}')^\ell = (\boldsymbol{Z})_\ell^\top (\boldsymbol{Z}')_\ell$ exactly. Let $(\boldsymbol{Z})_{<\ell}$ stack $(\boldsymbol{Z})_0, \ldots, (\boldsymbol{Z})_{\ell-1}$. For convenience, denote all polynomials at the current level ℓ with the marker "F," and polynomials at all levels below the current level with the marker "L."
3. Construct the second-moment matrix of the degree-ℓ features, $C_{FF} \equiv \mathbb{E}\big[(\boldsymbol{Z})_\ell (\boldsymbol{Z})_\ell^\top\big]$.
4. Similarly, construct $C_{FL} \equiv \mathbb{E}\big[(\boldsymbol{Z})_\ell (\boldsymbol{Z})_{<\ell}^\top\big]$ between the degree-ℓ features and all lower-degree ones, and $C_{LL} \equiv \mathbb{E}\big[(\boldsymbol{Z})_{<\ell} (\boldsymbol{Z})_{<\ell}^\top\big]$.
5. Project out all lower-degree features using the Schur complement, $\tilde{C}_\ell = C_{FF} - C_{FL} C_{LL}^{+} C_{FL}^\top$, where $^+$ is the pseudoinverse (C_{LL} is singular because reordered products repeat).
6. The subkernel K_ℓ's eigendecomposition is equivalent to a diagonalization of $\frac{c_\ell}{\ell!}\tilde{C}_\ell$.
7. The eigendecomposition of the original kernel K is the set of eigendecompositions of each subkernel:

$$\text{Eigendecomp}(K) \approx \text{Eigendecomp}(K_0) + \text{Eigendecomp}(K_1) + \ldots$$

Note on Hermite polynomials For Gaussian data, the level-ℓ eigenfunctions are Hermite polynomials of degree ℓ in the data's principal components (Karkada et al., 2026). ICL data is exactly Gaussian only for a single task ($T = 1$), where each label is a linear function of its input plus Gaussian noise; there, the subkernel expansion reduces to their Hermite structure. For $T > 1$, mixing over tasks makes the context non-Gaussian and correlates distinct positions through the shared task (the matrix C of Section E.3), so the eigenfunctions depart from Hermite polynomials. The exception is the query, which is Gaussian and independent of the context, so its factor remains Hermite (Proposition 1).

E.2 Potential proof sketch for the subkernel expansion framework

We have not proven the subkernel expansion framework in general; it is validated experimentally in Figure 7. Nonetheless, we sketch a possible proof below.

Proof sketch main idea. Let μ be any data measure with $\|z\|^2 \le R^2$, and let $K(z, z') = \sum_\ell \frac{c_\ell}{\ell!}(z^\top z')^\ell$ with $c_{\ell+1} R^2 \le \varepsilon c_\ell$. For every fixed level ℓ, the true kernel eigenvalues match the level-ℓ subkernel expansion eigenvalues (with the same indices) up to an additive $O(\varepsilon c_\ell R^{2\ell}/\ell!)$. This is a relative $1 + O(\varepsilon)$ for eigenvalues of that order. The true eigenspaces match the framework's eigenspaces up to an overlap error of $O(\varepsilon)$ divided by the relative spectral gap. For the Gaussian kernel, $c_{\ell+1}/c_\ell = \sigma^{-2}$, so $\varepsilon \approx R^2/\sigma^2$.

The argument bounds each eigenvalue from above and below; the eigenspace overlap then follows by standard perturbation theory.

Upper bound. Apply Courant–Fischer on the functions orthogonal to $\mathcal{P}_{<\ell}$ and to the top $j-1$ level-ℓ framework eigenfunctions. There, the lower subkernels vanish, level ℓ contributes exactly the framework's quadratic form, and the higher levels contribute at most $\sum_{m>\ell} c_m R^{2m}/m! = O(\varepsilon c_\ell R^{2\ell}/\ell!)$.

Lower bound. Assume the lower-level eigenvalues are of order $c_{\ell-1} R^{2(\ell-1)}/(\ell-1)!$, which by hypothesis is at least of order $c_\ell R^{2\ell}/(\varepsilon\, \ell!)$. The coupling between $\mathcal{P}_{<\ell}$ and the level-ℓ functions comes through K_ℓ and the higher levels, and is of order $c_\ell R^{2\ell}/\ell!$. Eliminating the lower block therefore shifts the eigenvalue by at most $O(\varepsilon c_\ell R^{2\ell}/\ell!)$.

E.3 Using the subkernel expansion framework with ICL data

Above, we introduced the subkernel expansion. Now, we apply it to the ICL linear regression setting to find the eigensystem beyond heuristics.

We begin $d = 1$ for readability, so each x_i is a scalar with variance σ_x^2, and we write $m_k = \frac{1}{T}\sum_{t=1}^{T} \beta_t^k$ for the raw moments of the pool.

Linear subkernel. The level-1 subkernel $K_1 = c_1 \boldsymbol{z}^\top \boldsymbol{z}'$ performs linear regression. Its eigenfunctions are the principal components of the data's second-moment matrix, with eigenvalues c_1 times those of the matrix. Every coordinate of $\boldsymbol{z}$ has mean zero, so no projection against the constant is needed, and the second-moment matrix is the covariance. Distinct positions are uncorrelated: $\mathrm{Cov}(x_i, x_j) = \mathrm{Cov}(x_i, y_j) = \mathrm{Cov}(y_i, y_j) = 0$ for $i \neq j$, and the query is uncorrelated with every in-context coordinate. The covariance is therefore block diagonal. It consists of an isolated query block, with eigenfunction x_{L+1} and eigenvalue $c_1\sigma_x^2$, and L identical 2×2 blocks, one per position:

$$\mathrm{Cov}\begin{pmatrix} x_i \\ y_i \end{pmatrix} = \begin{pmatrix} \mathbb{E}_{\mathcal{D}}[x_i^2] & \mathbb{E}_{\mathcal{D}}[x_i(\beta x_i + \epsilon\eta_i)] \\ \mathbb{E}_{\mathcal{D}}[x_i(\beta x_i + \epsilon\eta_i)] & \mathbb{E}_{\mathcal{D}}[(\beta x_i + \epsilon\eta_i)^2] \end{pmatrix} = \begin{pmatrix} \sigma_x^2 & m_1\sigma_x^2 \\ m_1\sigma_x^2 & m_2\sigma_x^2 + \epsilon^2 \end{pmatrix},$$

where the expectation over $\mathcal{D}$ averages β over the finite pool. Each block has eigenvalues and (unnormalized) eigenfunctions

$$\lambda_\pm = \frac{1}{2}\left(\sigma_x^2(1+m_2) + \epsilon^2 \pm \sqrt{\left(\sigma_x^2(1-m_2) - \epsilon^2\right)^2 + 4m_1^2\sigma_x^4}\right),$$
$$\phi_{\pm,i} \propto m_1\sigma_x^2\, x_i + (\lambda_\pm - \sigma_x^2)\, y_i,$$

so the subkernel eigenvalues $c_1\lambda_\pm$ are each L-fold degenerate. Of these linear eigenfunctions, only x_{L+1} carries a nonzero target coefficient (the mean $\tilde{\mu}$ of Table 2).

Quadratic subkernel. Degree-2 polynomial have nonzero means, so at level 2 we must project out the constant; in practice this means centering each monomial. Projection against level 1 is automatic, since every third moment vanishes under the symmetry $\boldsymbol{z} \mapsto -\boldsymbol{z}$. The multiplicities matter here. Since $(\boldsymbol{z}^\top \boldsymbol{z}')^2 = \sum_{\mu\nu} z_\mu z_\nu z'_\mu z'_\nu$, a product of two distinct coordinates appears twice, so its feature carries a factor $\sqrt{2}$. With that convention, the eigenvalues of K_2 are $c_2/2!$ times those of the feature covariance. The covariance splits into blocks:

1. $x_{L+1}^2 - \sigma_x^2$ with itself.
2. $\sqrt{2}\, x_{L+1}x_i$ with $\sqrt{2}\, x_{L+1}y_i$: the linear-subkernel block with a factor of x_{L+1} attached, giving eigenvalues $2\sigma_x^2\lambda_\pm$, each L-fold degenerate.
3. For each pair $i < j$, the 4×4 block of $\sqrt{2}\, x_ix_j$, $\sqrt{2}\, x_iy_j$, $\sqrt{2}\, y_ix_j$ and $\sqrt{2}\, y_iy_j$, of which there are $L(L-1)/2$ identical copies. Because both positions share the same task, this block is $2\,\mathbb{E}_\beta[B(\beta) \otimes B(\beta)]$, where $B(\beta)$ is the conditional covariance of (x_i, y_i) given β; it is not the Kronecker square of the pool-averaged linear block.

The remaining polynomial require more care. Define the centered single-position features

$$a_i = x_i^2 - \sigma_x^2, \qquad b_i = \sqrt{2}\,(x_iy_i - m_1\sigma_x^2), \qquad c_i = y_i^2 - (m_2\sigma_x^2 + \epsilon^2).$$

Let D be their 3×3 covariance at a single position, and C their cross-covariance between two distinct positions $i \neq j$. Neither depends on which positions are chosen. Given β, distinct positions are independent, so C is the pool covariance of the features' conditional means,

$$C = \mathrm{Cov}_\beta\big(\mathbb{E}[(a_i, b_i, c_i) \mid \beta]\big) = \sigma_x^4 \begin{pmatrix} 0 & 0 & 0 \\ 0 & 2\,\mathrm{Var}\,\beta & \sqrt{2}\,\mathrm{Cov}(\beta, \beta^2) \\ 0 & \sqrt{2}\,\mathrm{Cov}(\beta, \beta^2) & \mathrm{Var}\,\beta^2 \end{pmatrix},$$

with variances and covariances taken over the pool. In particular, $\mathbb{E}[b_ib_j] = 2\sigma_x^4(m_2 - m_1^2)$ vanishes only when every task in the pool is identical. Unlike the linear case, distinct positions are therefore correlated through the shared task, and the features span a single $3L \times 3L$ block, $I_L \otimes (D - C) + \mathbf{1}\mathbf{1}^\top \otimes C$. Its eigensystem is the union of the following:

1. $D + (L-1)C$, for functions summed over the context, $\sum_i (\cdot)_i$;
2. $D - C$, which is $(L-1)$-fold degenerate, for contrasts between positions, such as $b_i - b_j$.

The first sector contains the two eigenvalues that grow with L; its eigenfunctions are spanned by $\sum_i b_i$ and $\sum_i c_i$ and diagonalize the pool covariance of (β, β^2). The sum $\sum_i a_i$ does not grow, since the first row of C vanishes. These are the $d = 1$ versions of the families S, A and W of Section F.3, with β and β^2 as the task objects of S and A.

Cubic and higher subkernels. We do not track the higher subkernels in detail, but one property is key. The ICL-relevant function $\sum_i x_i y_i \, x_{L+1}$ lies in the context-sum sector, whose eigenvalue grows linearly in L through $C_{bb} = 2\sigma_x^4 \operatorname{Var} \beta$, and therefore needs only to be orthonormalized using the subkernel expansion framework.

In higher dimensional data The same procedure outlined above applies in d dimensions. Within a position, the label $y_i = \boldsymbol{\beta}^\top \boldsymbol{x}_i + \epsilon\eta_i$ now couples to every coordinate of $\boldsymbol{x}_i$, so the per-position blocks become $(d+1) \times (d+1)$. Across positions, the structure is as in one dimension: distinct positions are uncorrelated at level 1, and at level 2 and above they are correlated only through the shared task.

F Eigencoefficients of the ICL Linear Regression problem

F.1 Kernels are approximations of memorizing solutions

In the main text we used the fact that a kernel learner's eigencoefficients are those of the memorizing solution. This holds in any supervised setting, not only ICL: it is the standard statement that kernel regression targets the conditional mean under the training distribution. We state it in the form we use.

Lemma 1 (Kernel eigencoefficients are those of the memorizing solution)**.** *Let $\{\phi_i(\boldsymbol{z})\}$ be the ($\boldsymbol{z}$-measurable) eigenbasis of a PSD kernel K. As $f_{\mathrm{mem}}(\boldsymbol{z}) = \mathbb{E}[y|\boldsymbol{z}]$ is the projection of $y^\star$ onto $\boldsymbol{z}$-measurable functions, the eigencoefficients of the target and a memorizing predictor are mode-identical:*

$$v_i \equiv \mathbb{E}_{(\boldsymbol{z},y)\sim\mathcal{D}_{\mathrm{tr}}}[y\phi_i(\boldsymbol{z})] = \mathbb{E}_{(\boldsymbol{z},y)\sim\mathcal{D}_{\mathrm{tr}}}[f_{mem}(\boldsymbol{z})\phi_i(\boldsymbol{z})] \tag{33}$$

for all i. Therefore, an infinite-sample kernel will implement the memorizing solution exactly, with a finite-sample kernel implementing an approximation based on the kernel's largest eigenvalues.

Proof. Let $\mathcal{D}_{\mathrm{tr}}$ be the training measure on contexts, let $y^\star \in L^2$ be a target whose randomness is not exhausted by Z, and let $f_{\mathrm{mem}}(\boldsymbol{z}) \equiv \mathbb{E}[y^\star|\boldsymbol{z}]$ be the Bayes-optimal predictor under $\mathcal{D}_{\mathrm{tr}}$. For any orthonormal set $\{\phi_i\}$, the law of total expectation yields

$$\mathbb{E}_{(\boldsymbol{z},y)\sim\mathcal{D}_{\mathrm{tr}}}[y^\star, \phi_i] = \mathbb{E}\left[\mathbb{E}\left[y^\star\phi_i(\boldsymbol{z})|\boldsymbol{z}\right]\right] = \mathbb{E}\left[\phi_i(\boldsymbol{z})\mathbb{E}\left[y^\star|\boldsymbol{z}\right]\right] = \mathbb{E}_{(\boldsymbol{z},y)\sim\mathcal{D}_{\mathrm{tr}}}[f_{\mathrm{mem}}, \phi_i].$$

□

F.2 Whitened eigensystem

The main text takes $\Sigma_{\boldsymbol{x}} \propto I_d$. General $\Sigma_{\boldsymbol{x}}$ reduces to this case through the rescaling $\boldsymbol{\beta}^\top \boldsymbol{x} = (\Sigma_{\boldsymbol{x}}^{1/2}\boldsymbol{\beta})^\top(\Sigma_{\boldsymbol{x}}^{-1/2}\boldsymbol{x})$. This invertible linear map leaves the space of polynomials of each degree unchanged, so the eigencoefficients are those of the isotropic case with the tasks replaced by the colored tasks $\Sigma_{\boldsymbol{x}}^{1/2}\boldsymbol{\beta}$, and the cumulants that enter are those of the colored pool. The kernel itself is not whitened: its eigenvalues are still computed in raw coordinates and depend on $\Sigma_{\boldsymbol{x}}$.

We denote whitened and colored quantities with a tilde,

$$\tilde{\boldsymbol{x}} = \Sigma_{\boldsymbol{x}}^{-1/2}\boldsymbol{x}, \qquad \tilde{\boldsymbol{\beta}} = \Sigma_{\boldsymbol{x}}^{1/2}\boldsymbol{\beta}, \qquad \tilde{\mathcal{B}}_{\mathrm{tr}} = \mathcal{U}\left(\{\Sigma_{\boldsymbol{x}}^{1/2}\boldsymbol{\beta}^{(t)}\}_{t=1}^{T}\right), \tag{34}$$

and write $\tilde{\kappa}_\ell$ for the cumulants of $\tilde{\mathcal{B}}_{\mathrm{tr}}$; by multilinearity (Equation (14)), $\tilde{\kappa}_\ell$ is $\hat{\kappa}_\ell$ with every index contracted against $\Sigma_{\boldsymbol{x}}^{1/2}$. We write $\tilde{\mu}$, $\tilde{M}_2$ and $\tilde{\Sigma}$ for the mean, second moment, and covariance of $\tilde{\mathcal{B}}_{\mathrm{tr}}$, $(\tilde{s}_a, \boldsymbol{u}_a)$ for the eigenpairs of $\tilde{\Sigma}$, and Tr_k for the partial trace of Equation (15). Whitening gives $\tilde{\boldsymbol{x}}_i \sim \mathcal{N}(0, I_d)$, and the labels $y_i = \tilde{\boldsymbol{\beta}}^\top\tilde{\boldsymbol{x}}_i + \epsilon\eta_i$ are unchanged. We define the per-task label variance and its pool average,

$$\tau(\boldsymbol{\beta}) = \boldsymbol{\beta}^\top\Sigma_{\boldsymbol{x}}\boldsymbol{\beta} + \epsilon^2 = \|\tilde{\boldsymbol{\beta}}\|^2 + \epsilon^2, \qquad \bar{\tau} = \mathbb{E}_{\mathcal{B}_{\mathrm{tr}}}[\tau(\boldsymbol{\beta})] = \mathrm{Tr}\left(\tilde{M}_2\right) + \epsilon^2. \tag{35}$$

We collect the L in-context pairs, but not the query, into the context matrix $\boldsymbol{z}_{\mathrm{c}} = [\tilde{X}\ \boldsymbol{y}] \in \mathbb{R}^{L\times(d+1)}$, whose i-th row is $(\tilde{\boldsymbol{x}}_i^\top, y_i)$, and form its Gram matrix

$$M \equiv Z_{\mathrm{c}}^\top Z_{\mathrm{c}} = \begin{pmatrix} W & S \\ S^\top & A \end{pmatrix}, \qquad W = \sum_{i=1}^{L}\tilde{\boldsymbol{x}}_i\tilde{\boldsymbol{x}}_i^\top, \quad S = \sum_{i=1}^{L} y_i\tilde{\boldsymbol{x}}_i, \quad A = \sum_{i=1}^{L} y_i^2. \tag{36}$$

Proposition 1 (The target lives on polynomials in M)**.** *Given $\boldsymbol{\beta}$, the rows of $\boldsymbol{z}_\mathrm{c}$ are i.i.d. $\mathcal{N}(0, \Sigma_{\boldsymbol{z}|\boldsymbol{\beta}})$ with $\Sigma_{\boldsymbol{z}|\boldsymbol{\beta}} = \left(\begin{smallmatrix} I_d & \tilde{\boldsymbol{\beta}} \\ \tilde{\boldsymbol{\beta}}^\top & \tau(\boldsymbol{\beta}) \end{smallmatrix}\right)$, and the query $\tilde{\boldsymbol{x}}_{L+1} \sim \mathcal{N}(0, I_d)$ is independent of $\boldsymbol{z}_\mathrm{c}$. Hence, under the pool or under the prior, the density of $\boldsymbol{z}_\mathrm{c}$ depends on $\boldsymbol{z}_\mathrm{c}$ only through M, and it is invariant under $\boldsymbol{z}_\mathrm{c} \mapsto Q Z_\mathrm{c}$ for every $Q \in O(L)$ acting on context positions. A rotation-invariant kernel is invariant under the same map applied to both arguments, since it leaves $\boldsymbol{z}^\top \boldsymbol{z}' = \mathrm{Tr}\big(\boldsymbol{z}_\mathrm{c}^\top \boldsymbol{z}_\mathrm{c}'\big) + \boldsymbol{x}_{L+1}^\top \boldsymbol{x}_{L+1}'$ and $\|\boldsymbol{z}\|$ unchanged. It follows that:*

1. *The kernel's eigenfunctions can be chosen to be either $O(L)$-invariant or orthogonal to every $O(L)$-invariant function. The target is invariant, so only invariant eigenfunctions carry nonzero eigencoefficients, under the pool or the prior. The invariant polynomials of $\boldsymbol{z}_\mathrm{c}$ are exactly the polynomials in the entries of M.*
2. *Within the subkernel expansion (Definition 4), the degree-p eigenfunctions separate by Hermite degree in the query, because the query is Gaussian and independent of the context. The target is linear in $\tilde{\boldsymbol{x}}_{L+1}$, so it overlaps only eigenfunctions of Hermite degree one in the query.*

Every eigenfunction with a nonzero eigencoefficient is therefore of the form $\phi(M)^\top \tilde{\boldsymbol{x}}_{L+1}$, with ϕ a vector of polynomials in the entries of M. At data degree $p = 2k+1$, these polynomials have degree at most k in M.

Proof. For (i), the integral operator of K commutes with the unitary action of $O(L)$ on L^2, so each of its eigenspaces is $O(L)$-invariant and splits orthogonally into its invariant functions and their complement. An invariant target has zero inner product with the complement. That the invariant polynomials are the polynomials in M is the first fundamental theorem of invariant theory for $O(L)$, applied to the $d+1$ columns of $\boldsymbol{z}_\mathrm{c}$. For (ii), after the projections of Definition 4, a level-p feature of query degree j retains only the Hermite-degree-j part of its query factor. Hermite polynomials of different degree in $\tilde{\boldsymbol{x}}_{L+1}$ are orthogonal, and they are independent of the context. □

In particular, sums that are invariant under permutations of the context but not under $O(L)$, such as $\sum_i y_i^4$ or $\sum_i y_i^3 \tilde{\boldsymbol{x}}_i$, carry no coefficient of their own; their invariant parts lie in the span of polynomials in M. We call a monomial type

$$\psi_{(n_W, n_S, n_A)} = W^{\otimes n_W} \otimes S^{\otimes n_S} A^{n_A} \otimes \tilde{\boldsymbol{x}}_{L+1}, \qquad n_W + n_S + n_A = k, \tag{37}$$

a *family* of degree $p = 2k+1$. For example, the degree-3 families are $S \otimes \tilde{\boldsymbol{x}}_{L+1}$, $A\,\tilde{\boldsymbol{x}}_{L+1}$ and $W \otimes \tilde{\boldsymbol{x}}_{L+1}$, and there are six families at degree 5. Every family carries a single factor of the query. The query is independent of the context and has $\mathbb{E}[\tilde{\boldsymbol{x}}_{L+1} \tilde{\boldsymbol{x}}_{L+1}^\top] = I_d$, so perpendicularization (as is done in Section E) acts on the context factor alone. We write $\tilde{\Psi}_p = \tilde{\Psi}_p^\mathrm{c} \otimes \tilde{\boldsymbol{x}}_{L+1}$, where $\tilde{\Psi}_p^\mathrm{c}$ stacks the context factors of all degree-p families, and write $\tilde{\Psi}_p^{\mathrm{c}\perp}$ for its perpendicular version.

Definition 5 (Colored eigenfunctions, general $\Sigma_{\boldsymbol{x}}$)**.** Let $\Gamma_p \equiv \mathbb{E}\big[\tilde{\Psi}_p^{\mathrm{c}\perp}\, \tilde{\Psi}_p^{\mathrm{c}\perp\top}\big]$ be the Gram matrix of the degree-p context factors. It is singular because the tensor families contain repeated entries, so inverses are taken on its range. For any orthogonal matrix R_p,

$$\Phi_p \equiv R_p\, \Gamma_p^{-1/2}\, \tilde{\Psi}_p^{\mathrm{c}\perp}, \qquad \tilde{\phi}_{p,m,b} = \Phi_{p,m}\, \tilde{x}_{L+1,b} \tag{38}$$

are orthonormal under the training measure and span the degree-p eigenfunctions allowed by Proposition 1. Unless stated otherwise we take $R_p = I$; the kernel's own eigenbasis corresponds to the R_p given below.

The families of a single degree are not in general orthogonal to one another. At degree 3 they nearly are. The leading-order overlap between the S and A families is set by the pool covariance $\mathrm{Cov}_{\tilde{\mathcal{B}}_\mathrm{tr}}(\tilde{\boldsymbol{\beta}}, \|\tilde{\boldsymbol{\beta}}\|^2) = \mathrm{Tr}_1\, \tilde{\kappa}_3 + 2\tilde{\Sigma}\tilde{\mu}$. For a centered prior this is $O(T^{-1/2})$, and it vanishes for a centered, symmetric pool. The overlaps of W with S and A fall off as $L^{-1/2}$. From degree 5 on, each family's context factor concentrates on L^k times its task object $\tilde{\boldsymbol{\beta}}^{\otimes n_S} \tau^{n_A}$ (Section G), and the pool covariances between these task objects do not vanish as $L \to \infty$. No choice of families is then orthogonal, and the choice of R_p matters.

The kernel's eigenbasis within a degree. Definition 5 fixes the degree-p span but not the basis within it; the kernel's eigenfunctions correspond to one particular R_p. By Definition 4, the subkernel $K_p = \frac{c_p}{p!}(\boldsymbol{z}^\top \boldsymbol{z}')^p$ acts through the degree-p monomials of the raw context $\{(\boldsymbol{z})_p\}$, with lower degrees projected out; write $\widetilde{(\boldsymbol{z})}_{p,\mu}$ for these projected monomials. By Proposition 1, the span of Equation (38) is invariant under K_p, so step 6 of Definition 4 can be carried out inside it. In the basis $\tilde{\phi}_{p,m,b}$ (with $R_p = I$), K_p is the matrix

$$[\mathcal{K}_p]_{(m,b),(m',b')} = \frac{c_p}{p!} \sum_{\mu} \mathbb{E}\big[\tilde{\phi}_{p,m,b}\, \widetilde{(\boldsymbol{z})}_{p,\mu}\big]\, \mathbb{E}\big[\tilde{\phi}_{p,m',b'}\, \widetilde{(\boldsymbol{z})}_{p,\mu}\big], \tag{39}$$

which measures how strongly two orthonormal polynomials load onto the same monomials. Its eigenvectors give the rows of R_p, and its eigenvalues are the kernel eigenvalues. Because the query is independent of the context, $\mathcal{K}_p$ factorizes into a context part times $\Sigma_{\boldsymbol{x}}$: the query index is rotated into the eigenbasis of $\Sigma_{\boldsymbol{x}}$, and R_p acts on the context index m alone. Since the monomials are in raw coordinates, R_p depends on $\Sigma_{\boldsymbol{x}}$ even though the span does not.

F.3 The eigencoefficients

We start with a lemma for evaluating conditional expectations of the context factors.

Lemma 2 (Conditional moments of a context factor)**.** *For integers $a, b \geq 0$, the context factor $F_{a,b} \equiv \sum_{i=1}^{L} y_i^a \tilde{\boldsymbol{x}}_i^{\otimes b}$ is a sum of L i.i.d. terms given $\boldsymbol{\beta}$. Every joint cumulant of such factors given $\boldsymbol{\beta}$ is therefore L times the corresponding joint cumulant at a single position; in particular*

$$\mathbb{E}\big[F_{a,b} \,\big|\, \boldsymbol{\beta}\big] = L\, \mathbb{E}\big[y^a \tilde{\boldsymbol{x}}^{\otimes b} \,\big|\, \boldsymbol{\beta}\big], \qquad \mathrm{Cov}\big(F_{a,b} \,\big|\, \boldsymbol{\beta}\big) = L\, \mathrm{Cov}\big(y^a \tilde{\boldsymbol{x}}^{\otimes b} \,\big|\, \boldsymbol{\beta}\big). \tag{40}$$

The single-position cases used below are

$$\mathbb{E}\big[y\tilde{\boldsymbol{x}} \,\big|\, \boldsymbol{\beta}\big] = \tilde{\boldsymbol{\beta}}, \qquad \mathrm{Cov}\big(y\tilde{\boldsymbol{x}} \,\big|\, \boldsymbol{\beta}\big) = \tilde{\boldsymbol{\beta}}\tilde{\boldsymbol{\beta}}^\top + \tau I_d, \tag{41}$$

$$\mathbb{E}\big[y^2 \,\big|\, \boldsymbol{\beta}\big] = \tau, \qquad \mathrm{Var}\big(y^2 \,\big|\, \boldsymbol{\beta}\big) = 2\tau^2, \tag{42}$$

$$\mathbb{E}\big[\tilde{x}_a \tilde{x}_c \,\big|\, \boldsymbol{\beta}\big] = \delta_{ac}, \qquad \mathrm{Cov}\big(\tilde{x}_a \tilde{x}_c,\, \tilde{x}_{a'} \tilde{x}_{c'} \,\big|\, \boldsymbol{\beta}\big) = \delta_{aa'}\delta_{cc'} + \delta_{ac'}\delta_{ca'}, \tag{43}$$

$$\mathrm{Cov}\big(y\tilde{\boldsymbol{x}},\, y^2 \,\big|\, \boldsymbol{\beta}\big) = 2\tau\tilde{\boldsymbol{\beta}}, \qquad \mathrm{Cov}\big(y\tilde{x}_b,\, \tilde{x}_a \tilde{x}_c \,\big|\, \boldsymbol{\beta}\big) = \delta_{ab}\tilde{\beta}_c + \delta_{bc}\tilde{\beta}_a, \tag{44}$$

$$\mathrm{Cov}\big(y^2,\, \tilde{x}_a \tilde{x}_c \,\big|\, \boldsymbol{\beta}\big) = 2\tilde{\beta}_a \tilde{\beta}_c,$$

with $\tau = \tau(\boldsymbol{\beta})$.

Proof. The positions are independent given $\boldsymbol{\beta}$, and cumulants add over independent summands. For (41), $y\tilde{\boldsymbol{x}} = (\tilde{\boldsymbol{\beta}}^\top \tilde{\boldsymbol{x}})\tilde{\boldsymbol{x}} + \epsilon\eta\tilde{\boldsymbol{x}}$ has mean $\tilde{\boldsymbol{\beta}}$, and by Isserlis' theorem $\mathbb{E}[(\tilde{\boldsymbol{\beta}}^\top \tilde{\boldsymbol{x}})^2 \tilde{\boldsymbol{x}}\tilde{\boldsymbol{x}}^\top] = 2\tilde{\boldsymbol{\beta}}\tilde{\boldsymbol{\beta}}^\top + \|\tilde{\boldsymbol{\beta}}\|^2 I_d$, so the second moment is $2\tilde{\boldsymbol{\beta}}\tilde{\boldsymbol{\beta}}^\top + \tau I_d$ and the covariance follows. For (42), $y \,|\, \boldsymbol{\beta} \sim \mathcal{N}(0, \tau)$ gives $\mathrm{Var}(y^2 \,|\, \boldsymbol{\beta}) = 2\tau^2$. The remaining entries follow in the same way, since $(\tilde{\boldsymbol{x}}, y) \,|\, \boldsymbol{\beta}$ is jointly Gaussian with covariance $\Sigma_{z|\boldsymbol{\beta}}$ (Proposition 1). □

Task objects. By Lemma 2, each context sum concentrates on a function of the task: $W \approx L I_d$, $S \approx L\tilde{\boldsymbol{\beta}}$ and $A \approx L\tau$. The context factor of a degree-$p = 2k+1$ family therefore has conditional mean

$$\mathbb{E}\big[W^{\otimes n_W} \otimes S^{\otimes n_S} A^{n_A} \,\big|\, \boldsymbol{\beta}\big] = L^k\, I_d^{\otimes n_W} \otimes \tilde{\boldsymbol{\beta}}^{\otimes n_S} \tau^{n_A} + O(L^{k-1}), \tag{45}$$

where the $O(L^{k-1})$ corrections come from coincident positions. We call $f(\boldsymbol{\beta}) = I_d^{\otimes n_W} \otimes \tilde{\boldsymbol{\beta}}^{\otimes n_S} \tau^{n_A}$ the family's *task object*. We write $\boldsymbol{f}_p$ for the stacked task objects of all degree-p families, $\boldsymbol{f}_p^\perp$ for their residual after regression on all lower-degree task objects under the pool, and

$$G_p^\perp \equiv \mathrm{Cov}_{\tilde{\mathcal{B}}_{\mathrm{tr}}}\big(\boldsymbol{f}_p^\perp\big). \tag{46}$$

Directions with $G_p^\perp \neq 0$ are *task modes*. Directions with $G_p^\perp = 0$ are *noise modes*: the W families, and trace components such as $\mathrm{Tr}(S \otimes S) = \|S\|^2$, whose task objects repeat lower-degree ones.

Theorem 1 (The eigencoefficient-cumulant correspondence)**.** *Let $\tilde{\phi}_{p,m,b} = \Phi_{p,m} \tilde{x}_{L+1,b}$ be the orthonormal polynomials of Definition 5, and let $\boldsymbol{q}_p(\boldsymbol{\beta}) \equiv \mathbb{E}\big[\tilde{\Psi}_p^{\mathrm{c}\perp} \,\big|\, \boldsymbol{\beta}\big]$.*

1. Exact form. *The eigencoefficients are* $v_{1,b} = \tilde{\mu}_b$ *and, for* $p \geq 3$,

$$v_{p,\cdot,b} = R_p\, \Gamma_p^{-1/2}\, V_{p,b}, \qquad V_{p,b} = \mathbb{E}\big[\tilde{\beta}_b\, \tilde{\Psi}_p^{\mathrm{c}\perp}\big] = \mathrm{Cov}_{\tilde{\mathcal{B}}_{\mathrm{tr}}}\big(\tilde{\beta}_b,\ \boldsymbol{q}_p(\boldsymbol{\beta})\big). \tag{47}$$

2. Scaling with L. *Degree-3 task-mode coefficients are* $O(1)$. *For* $p = 2k+1 \geq 5$, $V_{p,b} = O(L^{k-1})$ *while* $\Gamma_p = L^{2k} G_p^{\perp} + O(L^{2k-1})$, *so task-mode coefficients are* $O(L^{-1})$ *and noise-mode coefficients are* $O(L^{-1/2})$.

3. Cumulant content. *At degree* $2k+1$, *the numerator of a family has at most* $k+1$ *free task indices, and only* $S^{\otimes k}$ *attains this bound. The cumulant* $\tilde{\kappa}_\omega$ *therefore first appears with all indices free at degree* $2\omega - 1$, *in the* $S^{\otimes(\omega-1)}$ *block; every lower degree contains it only through partial traces. The diagonal block of* Γ_p *for* $S^{\otimes(\omega-1)}$ *involves pool moments up to order* $2\omega - 2$, *and the full* Γ_p, *through the* A *families, involves moments up to order* $4\omega - 4$.

Proof. (i). By Lemma 1, $v_{p,m,b} = \mathbb{E}[y^\star \tilde{\phi}_{p,m,b}]$. The query is independent of the context and $\mathbb{E}[y_{L+1}\tilde{\boldsymbol{x}}_{L+1} \,|\, \boldsymbol{\beta}] = \tilde{\boldsymbol{\beta}}$, so $v_{p,m,b} = \mathbb{E}[\tilde{\beta}_b\, \Phi_{p,m}]$, and Equation (38) gives the first equality. Conditioning on $\boldsymbol{\beta}$ gives $V_{p,b} = \mathbb{E}_{\tilde{\mathcal{B}}_{\mathrm{tr}}}[\tilde{\beta}_b \boldsymbol{q}_p]$. For $p \geq 3$, $\tilde{\Psi}_p^{\mathrm{c}\perp}$ is orthogonal to the constant (the degree-1 context factor), so $\mathbb{E}[\boldsymbol{q}_p] = 0$ and this expectation is a covariance. For $p = 1$ the context factor is 1, and $v_{1,b} = \tilde{\mu}_b$.

(ii). By Lemma 2, the covariance between context factors of degrees $2k+1$ and $2k'+1$ is $L^{k+k'}$ times the pool covariance of their task objects, up to $O(L^{k+k'-1})$. At leading order, the perpendicularization of Definition 3 is therefore the regression of $\boldsymbol{f}_p$ on the lower-degree task objects under the pool, so $\boldsymbol{q}_p = L^k \boldsymbol{f}_p^{\perp} + O(L^{k-1})$ and $\Gamma_p = L^{2k} G_p^{\perp} + O(L^{2k-1})$. At $p = 3$, the only lower-degree task object is the constant, so $V_{3,b} = L\,\mathrm{Cov}(\tilde{\beta}_b, \boldsymbol{f}_3) + O(1)$ and the task-mode coefficients are $O(1)$. For $p \geq 5$, the lower degrees include the S family, whose task object is $\tilde{\boldsymbol{\beta}}$ itself. The residual $\boldsymbol{f}_p^{\perp}$ is then uncorrelated under the pool with every affine function of $\tilde{\boldsymbol{\beta}}$, so $\mathrm{Cov}(\tilde{\beta}_b, \boldsymbol{f}_p^{\perp}) = 0$ and the $O(L^k)$ part of $V_{p,b}$ vanishes. What survives is $O(L^{k-1})$. It arises because the data-space regression is onto the context factors rather than onto their task objects. For instance, S has conditional covariance $L(\tilde{\boldsymbol{\beta}}\tilde{\boldsymbol{\beta}}^\top + \tau I_d)$ by (41), which attenuates its regression coefficient by a relative $O(L^{-1})$ and leaves part of the linear component of $\boldsymbol{f}_p$ in place. Dividing by $\Gamma_p^{1/2}$, which is $O(L^k)$ on task modes and $O(L^{k-1/2})$ on noise modes, gives the stated orders.

(iii). A factor W contributes I_d and a factor A contributes the scalar τ, so the task object of (n_W, n_S, n_A) has n_S free task indices, and the target adds the index b. Contracting against I_d, τ or the regression coefficients adds no free indices, so $V_{p,b}$ has at most $n_S + 1 \leq k+1$ of them, with equality only for $S^{\otimes k}$. A pool moment of order ω with all of its indices free, and hence $\tilde{\kappa}_\omega$, requires $k+1 \geq \omega$. It can therefore first appear at degree $2\omega - 1$, and only in the $S^{\otimes(\omega-1)}$ block. It does appear there: the attenuation term of (ii) contains a term proportional to $\mathrm{Cov}_{\tilde{\mathcal{B}}_{\mathrm{tr}}}(\tilde{\boldsymbol{\beta}}^{\otimes(\omega-1)}, \tilde{\boldsymbol{\beta}})$, as in the example below. The diagonal block of Γ_p for $S^{\otimes(\omega-1)}$ pairs two task objects of degree $\omega - 1$, giving moments up to order $2\omega - 2$. The A families have task objects of degree up to $2\omega - 2$ in $\tilde{\boldsymbol{\beta}}$, so the full Γ_p involves moments up to order $4\omega - 4$. □

The above eigencoefficient equation is highly opaque, and will remain so for full completeness. Nonetheless, we detail the first few eigencoefficients for lower-degree polynomials.

Example: the SS block at degree 5. Take a centered pool with $\mathrm{Tr}_1\, \tilde{\kappa}_3 = 0$. Then S is uncorrelated with A, W and the constant, and only the regression of $S \otimes S$ on S contributes to $V_{SS,b}$. By Lemma 2,

$$\mathbb{E}[S \otimes S \,|\, \boldsymbol{\beta}] = L(L+1)\, \tilde{\boldsymbol{\beta}}\tilde{\boldsymbol{\beta}}^\top + L\tau I_d, \qquad \mathrm{Cov}(S) = L(L+1)\tilde{\Sigma} + L\bar{\tau} I_d$$
$$\mathbb{E}[S^{\otimes 3}] = (L^3 + 3L^2)\, \tilde{\kappa}_3 + O(L).$$

The regression coefficient of $S \otimes S$ on S is $\mathbb{E}[S^{\otimes 3}]\,\mathrm{Cov}(S)^{-1}$, and $\mathrm{Cov}(\tilde{\beta}_b, \mathbb{E}[S \,|\, \boldsymbol{\beta}]) = L\tilde{\Sigma}_{b\cdot}$, so

$$V_{SS,b} = L(L+1)\, \tilde{\kappa}_{3,b\cdot\cdot} - (L^3+3L^2)\big[\big((L+1)\tilde{\Sigma} + \bar{\tau} I_d\big)^{-1}\tilde{\Sigma}\big]_{bb'}\, \tilde{\kappa}_{3,b'\cdot\cdot} = L\big[(\bar{\tau}\tilde{\Sigma}^{-1} - I_d)\, \tilde{\kappa}_3\big]_{b\cdot\cdot} + O(1).$$

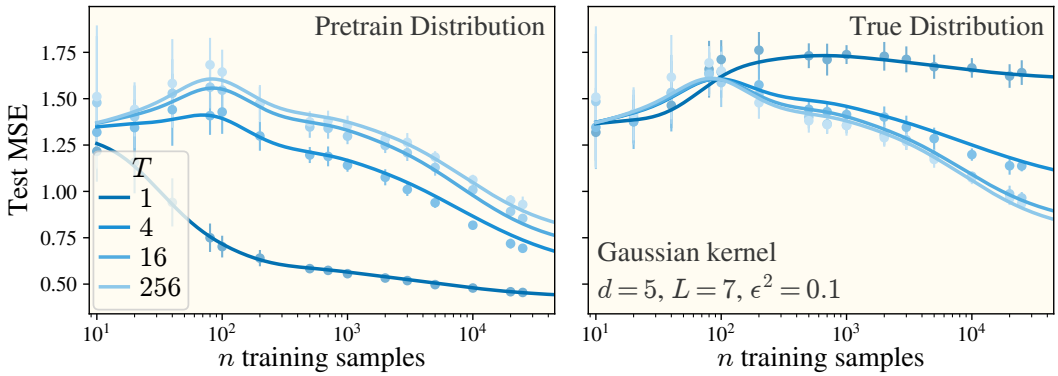


Figure 8: Gaussian kernel learning curves on ICL Regression data. For four different settings (colors) of the task diversity T, we compute the expected test MSE as a function the the number of training samples n, and compare it to empirical errors obtained across 20 trials. Theoretical learning curve equations are provided in Section D, with validation of the eigensystem the equations are built off of provided in Sections E and F. Our data uses a Gaussian kernel with kernel width $\sigma_{\text{ker}} = 3.0$, $d = 5, L = 7, \epsilon^2 = 0.1$ where our data is drawn $\boldsymbol{x} \sim \mathcal{N}(0, I_d/(dL)$ Additionally, as $T \to \infty$, we find that the errors on the pretraining distribution match those of the true distribution, as expected. Further experimental details are in Section B.

The $O(L^2)$ terms cancel as in (ii). The surviving $\bar{\tau}\tilde{\Sigma}^{-1}$ is the per-position noise-to-signal ratio of S, which sets the attenuation. With $\Gamma_{SS} = L^4 G_5^{\perp} + O(L^3)$, this gives the SS entry of Table 2.

Equation (47) is exact, but its content is easiest to read family by family. Table 2 lists the leading-order coefficients through degree seven.

Degree p	family	task object	eigencoefficient (leading order)
1	1	1	$\tilde{\mu}_b$
3	S	$\tilde{\boldsymbol{\beta}}$	$\tilde{s}_a / \sqrt{(1 + L^{-1})\tilde{s}_a + \bar{\tau}/L}$
	A	τ	$c_b / \sqrt{\mathrm{Var}\,\tau + 2\,\mathbb{E}\tau^2/L}, \quad c = \mathrm{Tr}_1\,\tilde{\kappa}_3 + 2\tilde{\Sigma}\tilde{\mu}$
	W	I_d	0 (noise mode)
5	SS	$\tilde{\boldsymbol{\beta}}^{\otimes 2}$	$L^{-1}(G_5^{\perp})^{-1/2}(\bar{\tau}\tilde{\Sigma}^{-1} - I_d)\tilde{\kappa}_3$
	SA	$\tau\tilde{\boldsymbol{\beta}}$	$L^{-1}(G_5^{\perp})^{-1/2}\left(-(2\,\mathrm{Tr}\,\tilde{\Sigma}^2\, I_d + 4\tilde{\Sigma}^2) + \dots\right)$
	AA	τ^2	$O(L^{-1})$; odd moments
	W families	$\dots$	$O(L^{-1/2})$ (noise modes)
7	SSS	$\tilde{\boldsymbol{\beta}}^{\otimes 3}$	$L^{-1}(G_7^{\perp})^{-1/2}\,\frac{d}{d+2}\,\frac{\epsilon^2}{\tilde{s}}\,\tilde{\kappa}_4$
	SAA	$\tau^2\tilde{\boldsymbol{\beta}}$	$\propto L^{-1}\ \mathrm{Tr}_2\,\tilde{\kappa}_6$
	SSA, AAA	$\tau\tilde{\boldsymbol{\beta}}^{\otimes 2}$, τ^3	odd moments; zero for symmetric pools
	W families	lower-degree objects	$O(L^{-1/2})$ (noise modes)

Table 2: Leading-order eigencoefficients through degree seven, in the symmetric basis $R_p = I$ (Definition 5). $G_p^{\perp}$ is the pool covariance of the degree-p task objects after regression on lower-degree ones, joint across families. Degree-3 rows treat the S, A and W blocks as uncoupled and are written in the eigenbasis of $\tilde{\Sigma}$. The S row is exact at every L when $\tilde{\mu} = 0$ and $\mathrm{Tr}_1\,\tilde{\kappa}_3 = 0$, and the A row is first order in c. Degree-5 rows are leading order in $1/L$. The SS row assumes $\tilde{\mu} = 0$ and $\mathrm{Tr}_1\,\tilde{\kappa}_3 = 0$; otherwise terms in $\tilde{\mu}$ and $\mathrm{Tr}_1\,\tilde{\kappa}_3$ are added. The SA row shows the Gaussian part for a centred pool. Degree-7 rows are leading order in $1/L$ for an isotropic centred pool ($\tilde{\Sigma} = \tilde{s}\, I_d$), at first order in the deviation from a Gaussian. Noise modes are the W families and trace components such as $\mathrm{Tr}(S \otimes S)$ (part of the SS family), whose task objects repeat lower-degree ones.

F.4 Validation of the eigencoefficients

To validate our theoretically derived eigencoefficients as the correct set used by the kernel learner, we use the theoretically-derived learning curve (test error vs number of samples) plotted against empirically found errors. We do not directly compare the eigencoefficients that are found empirically

to those theoretically derived, as we note that the "interesting region" of the eigencoefficients (the portion of the spectrum beyond the degree 1 terms) is partially affected by noise, with us not having enough memory to obtain clean empirical estimates. Therefore, we test our eigencoefficients in a different manner. Should our eigensystem, including the eigencoefficients, be correct, the learning curve (test error given number of samples) can be theoretically predicted, and will match well with empirical findings. In Figure 8, we do said experiment and find excellent agreement between our pure-theory curves and the empirically found kernel regression errors.

G TASK SPACE LEARNING

In Sections E and F, we described the kernel's eigensystem in the raw data space of Z. The same results can be read in *task space*: averaging the predictor over the in-context inputs and noise, $\boldsymbol{x}$ and η, gives the task the kernel predicts on average for a given latent $\boldsymbol{\beta}$. Within the eigenframework this average is exact; the approximations below enter only through the large-L forms of the task images.

G.1 EIGENFUNCTIONS IN TASK SPACE

Write each orthonormal polynomial of Definition 5 as $\tilde{\phi}_{p,m,b}(Z) = \Phi_{p,m}\,\tilde{x}_{L+1,b}$, where $\Phi_{p,m}$ depends only on the context, and define its *task image*

$$\chi_{p,m}(\boldsymbol{\beta}) \equiv \mathbb{E}_{\tilde{\boldsymbol{x}},\eta}\big[\Phi_{p,m}\,\big|\,\boldsymbol{\beta}\big], \qquad \text{so that} \qquad \boldsymbol{\chi}_p(\boldsymbol{\beta}) = R_p\,\Gamma_p^{-1/2}\,\boldsymbol{q}_p(\boldsymbol{\beta}), \tag{48}$$

with $\boldsymbol{q}_p$ as in Theorem 1. By Theorem 1(i), the eigencoefficients are task-space overlaps. By the law of total covariance, the task images are orthonormal up to the within-context noise:

$$v_{p,m,b} = \mathbb{E}_{\tilde{\mathcal{B}}_{\rm tr}}\big[\tilde{\beta}_b\,\chi_{p,m}\big], \qquad \mathbb{E}_{\tilde{\mathcal{B}}_{\rm tr}}\big[\chi_{p,m}\chi_{p',m'}\big] = \delta_{pp'}\delta_{mm'} - \mathbb{E}_{\tilde{\mathcal{B}}_{\rm tr}}\big[\mathrm{Cov}(\Phi_{p,m},\Phi_{p',m'}\,|\,\boldsymbol{\beta})\big]. \tag{49}$$

At large L, the task images simplify. By Theorem 1(ii), $\boldsymbol{q}_p = L^k\boldsymbol{f}_p^{\perp} + O(L^{k-1})$ and $\Gamma_p = L^{2k}G_p^{\perp} + O(L^{2k-1})$, with the task objects $\boldsymbol{f}_p$ and their pool covariance $G_p^{\perp}$ as defined in Section F.3. On task modes, therefore,

$$\boldsymbol{\chi}_p(\boldsymbol{\beta}) \approx R_p\,(G_p^{\perp})^{-1/2}\,\boldsymbol{f}_p^{\perp}(\boldsymbol{\beta}), \tag{50}$$

while on noise modes $\boldsymbol{f}_p^{\perp} = 0$ and the task images vanish as $O(L^{-1/2})$. At large L, the task images are thus orthonormal polynomials of $\tilde{\boldsymbol{\beta}}$ under the pool measure. At degree $p = 2k+1$, the new objects are $\tilde{\boldsymbol{\beta}}^{\otimes n_S}\tau^{n_A}$ with $n_S + n_A = k$, of degree k to $2k$ in $\tilde{\boldsymbol{\beta}}$. As in data space, each degree is regressed on all lower degrees and then orthonormalized as a block through $G_p^{\perp}$.

G.2 EIGENCOEFFICIENTS IN TASK SPACE

Write the predictor as $f(Z) = \hat{\tilde{\boldsymbol{\beta}}}^{\top}\tilde{\boldsymbol{x}}_{L+1}$, where $\hat{\tilde{\boldsymbol{\beta}}}$ is the predicted colored task. Take $\Sigma_{\boldsymbol{x}} \propto I_d$, as in all our experiments, so the learnability $\mathcal{L}_{p,m}$ of a mode does not depend on the query index. Averaging over the context given $\boldsymbol{\beta}$ gives

$$\mathbb{E}_{\tilde{\boldsymbol{x}},\eta}\big[\hat{\tilde{\beta}}_b\,\big|\,\boldsymbol{\beta}\big] = \sum_{p,m}\mathcal{L}_{p,m}\,v_{p,m,b}\,\chi_{p,m}(\boldsymbol{\beta}). \tag{51}$$

Here v and χ must be taken in the kernel's own basis (Equation (39)), since that is the basis in which learnabilities are assigned. For general $\Sigma_{\boldsymbol{x}}$, $\mathcal{L}$ also depends on the query index through the eigenvalues of $\Sigma_{\boldsymbol{x}}$, with b taken in that eigenbasis. Once a degree is fully learned ($\mathcal{L}_{p,m} = 1$ for all m), its contribution is basis-independent:

$$\sum_m v_{p,m,b}\,\chi_{p,m}(\boldsymbol{\beta}) = V_{p,b}^{\top}\,\Gamma_p^{-1}\,\boldsymbol{q}_p(\boldsymbol{\beta}) \;\approx\; L^{-k}\,V_{p,b}^{\top}\,(G_p^{\perp})^{-1}\boldsymbol{f}_p^{\perp}(\boldsymbol{\beta}). \tag{52}$$

The terms arrive in order of degree. At degree 1, the kernel learns the pool mean $\tilde{\mu}$. At degree 3, for $\tilde{\mu} = 0$ and $c = \mathrm{Tr}_1\,\tilde{\kappa}_3 + 2\tilde{\Sigma}\tilde{\mu} = 0$, the S modes contribute

$$\sum_a \frac{L\tilde{s}_a}{(L+1)\tilde{s}_a + \bar{\tau}}\,\boldsymbol{u}_a\boldsymbol{u}_a^{\top}\tilde{\boldsymbol{\beta}},$$

a shrinkage estimate of the true task. It recovers $\tilde{\boldsymbol{\beta}}$ in directions where $\tilde{s}_a \gg \bar{\tau}/L$, and not at all in directions the pool does not span ($\tilde{s}_a = 0$, which is unavoidable when $T \le d$). A nonzero c adds the A mode, a scalar function of $\tau(\boldsymbol{\beta})$ along the fixed direction c. From degree 5 on, the learner adds nonlinear functions of $\tilde{\boldsymbol{\beta}}$ of degree k to $2k$, each with coefficient $O(L^{-1})$. By Theorem 1(iii), these bring in the pool's skewness at degree 5, its kurtosis at degree 7, and so on. Table 3 lists the task images and coefficients through degree seven.

Degree p	family	$\chi_{p,m}(\boldsymbol{\beta})$	$v_{p,m,b}$
1	1	1	$\tilde{\mu}_b$
3	S	$\dfrac{\boldsymbol{u}_a^\top \tilde{\boldsymbol{\beta}}}{\sqrt{(1+L^{-1})\tilde{s}_a + \bar{\tau}/L}}$	$\dfrac{\tilde{s}_a}{\sqrt{(1+L^{-1})\tilde{s}_a + \bar{\tau}/L}}$
3	A	$\dfrac{\tau - \bar{\tau}}{\sqrt{\operatorname{Var}\tau + 2\,\mathbb{E}\tau^2/L}}$	$\dfrac{c_b}{\sqrt{\operatorname{Var}\tau + 2\,\mathbb{E}\tau^2/L}}$
3	W	0	0 (noise mode)
5	SS	$(G_5^\perp)^{-1/2}\big(\tilde{\boldsymbol{\beta}}^{\otimes 2}\big)^\perp$	$L^{-1}(G_5^\perp)^{-1/2}(\bar{\tau}\tilde{\Sigma}^{-1} - I_d)\tilde{\kappa}_3$
5	SA	$(G_5^\perp)^{-1/2}\big(\tau\tilde{\boldsymbol{\beta}}\big)^\perp$	$L^{-1}(G_5^\perp)^{-1/2}\left(-(2\operatorname{Tr}\tilde{\Sigma}^2\, I_d + 4\tilde{\Sigma}^2) + \dots\right)$
5	AA	$(G_5^\perp)^{-1/2}\big(\tau^2\big)^\perp$	$O(L^{-1})$; odd moments
5	W families	$O(L^{-1/2})$	$O(L^{-1/2})$ (noise modes)
7	SSS	$(G_7^\perp)^{-1/2}\big(\tilde{\boldsymbol{\beta}}^{\otimes 3}\big)^\perp$	$L^{-1}(G_7^\perp)^{-1/2}\frac{d}{d+2}\frac{\epsilon^2}{\tilde{s}}\tilde{\kappa}_4$
7	SSA	$(G_7^\perp)^{-1/2}\big(\tau\tilde{\boldsymbol{\beta}}^{\otimes 2}\big)^\perp$	odd moments
7	SAA	$(G_7^\perp)^{-1/2}\big(\tau^2\tilde{\boldsymbol{\beta}}\big)^\perp$	$\propto L^{-1}\ \operatorname{Tr}_2\tilde{\kappa}_6$
7	AAA	$(G_7^\perp)^{-1/2}\big(\tau^3\big)^\perp$	odd moments
7	W families	$O(L^{-1/2})$	$O(L^{-1/2})$ (noise modes)

Table 3: Task images and eigencoefficients through degree seven, in the symmetric basis $R_p = I$. $(\cdot)^\perp$ is the regression on all lower-degree task objects under the pool measure, and $(G_p^\perp)^{-1/2}$ is the symmetric inverse square root of the joint degree-p pool covariance. The family column therefore labels blocks of $\boldsymbol{f}_p^\perp$, which this rotation mixes unless $G_p^\perp$ is block diagonal. A family with n_S factors of S has a task image of tensor rank n_S; its coefficient carries $n_S + 1$ indices, of which n_S contract against the task image and one is the output index b. Degree-3 rows are exact at every L and are written in the eigenbasis of $\tilde{\Sigma}$: the S row when $\tilde{\mu} = 0$ and $\operatorname{Tr}_1\tilde{\kappa}_3 = 0$, and the A row at first order in $c = \operatorname{Tr}_1\tilde{\kappa}_3 + 2\tilde{\Sigma}\tilde{\mu}$. Degree-5 rows are leading order in $1/L$; the SS row assumes $\tilde{\mu} = 0$ and $\operatorname{Tr}_1\tilde{\kappa}_3 = 0$, and the SA row shows the Gaussian part for a centred pool. Degree-7 rows are leading order in $1/L$ for an isotropic centred pool ($\tilde{\Sigma} = \tilde{s}\, I_d$), at first order in the deviation from a Gaussian. χ is quoted at $L \to \infty$ except at degree 3.

H ORACLE EXPANSIONS

Throughout this appendix we work in raw coordinates, with $W \equiv \sum_{i=1}^L \boldsymbol{x}_i\boldsymbol{x}_i^\top$, $S \equiv \sum_{i=1}^L y_i\boldsymbol{x}_i$, $A \equiv \sum_{i=1}^L y_i^2$, and $\Delta W \equiv W - L\Sigma_{\boldsymbol{x}}$. These are the unwhitened versions of the sums in Section A. We write $\Sigma_{\boldsymbol{\beta}}$ and $\boldsymbol{\mu}_{\boldsymbol{\beta}}$ for the prior's covariance and mean, and $\hat{\mu}_{\boldsymbol{\beta}}, \hat{\Sigma}_{\boldsymbol{\beta}}, \hat{\kappa}_\ell$ for the cumulants of the pool $\mathcal{B}_{\mathrm{tr}}$.

Both oracles are Bayes-optimal under the *same* likelihood, $y_i \mid \boldsymbol{\beta} \sim \mathcal{N}(\boldsymbol{\beta}^\top\boldsymbol{x}_i, \epsilon^2)$; they differ only in the prior they place on $\boldsymbol{\beta}$. The ridge takes the population Gaussian $\mathcal{N}(\boldsymbol{\mu}_{\boldsymbol{\beta}}, \Sigma_{\boldsymbol{\beta}})$, and the dMMSE takes the uniform measure on the finite pool $\mathcal{B}_{\mathrm{tr}}$. Everything in this appendix follows from that single difference.

H.1 THE GENERALIZING SOLUTION

For $\boldsymbol{\beta} \sim \mathcal{N}(\boldsymbol{\mu}_{\boldsymbol{\beta}}, \Sigma_{\boldsymbol{\beta}})$, the Bayes-optimal predictor is $\hat{f}_{\mathrm{ridge}} = \boldsymbol{\beta}_{\mathrm{ridge}}^\top\boldsymbol{x}_{L+1}$, where $\boldsymbol{\beta}_{\mathrm{ridge}}$ is ridge regression with matrix parameter $\epsilon^2\Sigma_{\boldsymbol{\beta}}^{-1}$:

$$\boldsymbol{\beta}_{\mathrm{ridge}} = \left(W + \epsilon^2\Sigma_{\boldsymbol{\beta}}^{-1}\right)^{-1}\left(S + \epsilon^2\Sigma_{\boldsymbol{\beta}}^{-1}\boldsymbol{\mu}_{\boldsymbol{\beta}}\right). \tag{53}$$

The main text takes $\boldsymbol{\mu}_{\boldsymbol{\beta}} = 0$; we keep the general form for Section H.4.

H.2 THE MEMORIZING SOLUTION

For $\boldsymbol{\beta}$ uniform on $\mathcal{B}_{\rm tr}$, the Bayes-optimal predictor is $\hat{f}_{\rm mem} = \hat{\boldsymbol{\beta}}_{\rm mem}^\top \boldsymbol{x}_{L+1}$, with the posterior mean over the pool

$$\hat{\boldsymbol{\beta}}_{\rm mem} = \sum_{t=1}^{T} w_t\, \boldsymbol{\beta}^{(t)}, \qquad w_t = \frac{e^{\ell_t}}{\sum_s e^{\ell_s}}, \qquad \ell_t = \frac{1}{\epsilon^2}\Big(\boldsymbol{\beta}^{(t)\top} S - \tfrac{1}{2}\,\boldsymbol{\beta}^{(t)\top} W \boldsymbol{\beta}^{(t)}\Big). \tag{54}$$

This is Equation (3), after expanding $\sum_i (y_i - \boldsymbol{\beta}^\top \boldsymbol{x}_i)^2 = A - 2\boldsymbol{\beta}^\top S + \boldsymbol{\beta}^\top W \boldsymbol{\beta}$ and cancelling A, which is common to every task. Two consequences are used below. First, both oracles depend on the context only through W and S; the sum A enters the eigencoefficients only through its correlation with functions of W and S. Second, $\hat{\boldsymbol{\beta}}_{\rm mem} = \epsilon^2 \nabla_S \log \sum_t e^{\ell_t}$. A Taylor series of the dMMSE in S therefore generates cumulants of the pool reweighted by $e^{-\boldsymbol{\beta}^\top W \boldsymbol{\beta}/2\epsilon^2}$, which at $W \approx L\Sigma_{\boldsymbol{x}}$ is strongly tilted, rather than cumulants of the pool itself. The pool's own cumulants enter only after projection onto the orthonormal basis (Theorem 1).

H.3 WHY NO FINITE TRUNCATION OF THE ORACLE EXPANSIONS YIELDS THE EIGENCOEFFICIENTS

It is tempting to skip choosing the eigenfunction basis and instead Taylor-expand the oracles directly. This does not work: reasonable expansions identify the modes an oracle occupies, but truncating at any finite order misstates their coefficients, by arbitrary factors and in some cases all the way to zero. We record two examples as a warning.

Ridge. With $r_j \equiv y_j - \boldsymbol{\mu}_{\boldsymbol{\beta}}^\top \boldsymbol{x}_j$ and $\boldsymbol{A} \equiv \Big(L\Sigma_{\boldsymbol{x}} + \epsilon^2 \Sigma_{\boldsymbol{\beta}}^{-1}\Big)^{-1}$, so that $W + \epsilon^2 \Sigma_{\boldsymbol{\beta}}^{-1} = \boldsymbol{A}^{-1} + \Delta W$, the Neumann series gives

$$\begin{aligned}
\hat{f}_{\rm ridge} &= \boldsymbol{\mu}_{\boldsymbol{\beta}}^\top \boldsymbol{x}_{L+1} + \boldsymbol{x}_{L+1}^\top \Big(W + \epsilon^2 \Sigma_{\boldsymbol{\beta}}^{-1}\Big)^{-1} \sum_{j=1}^{L} r_j \boldsymbol{x}_j \\
&= \boldsymbol{\mu}_{\boldsymbol{\beta}}^\top \boldsymbol{x}_{L+1} + \sum_{m \geq 0} (-1)^m\, \boldsymbol{x}_{L+1}^\top \boldsymbol{A} (\Delta W \boldsymbol{A})^m \sum_{j=1}^{L} r_j \boldsymbol{x}_j \\
&= \boldsymbol{\mu}_{\boldsymbol{\beta}}^\top \boldsymbol{x}_{L+1} + \sum_j r_j\, \boldsymbol{x}_j^\top \boldsymbol{A}\, \boldsymbol{x}_{L+1} \\
&\qquad - \Big(\sum_{i,j} r_j\, (\boldsymbol{x}_j^\top \boldsymbol{A} \boldsymbol{x}_i)(\boldsymbol{x}_i^\top \boldsymbol{A} \boldsymbol{x}_{L+1}) - L \sum_j r_j\, \boldsymbol{x}_j^\top \boldsymbol{A} \Sigma_{\boldsymbol{x}} \boldsymbol{A}\, \boldsymbol{x}_{L+1}\Big) + \dots,
\end{aligned} \tag{55}$$

which converges since $\|\boldsymbol{A}\, \Delta W\| = O(L^{-1/2})$. The first line uses $S - W\boldsymbol{\mu}_{\boldsymbol{\beta}} = \sum_j r_j \boldsymbol{x}_j$.

Take $\Sigma_{\boldsymbol{x}} \propto I_d$ and $\boldsymbol{\mu}_{\boldsymbol{\beta}} = 0$, and evaluate under the prior, so that $(\tilde{s}_a, \boldsymbol{u}_a)$ are the eigenpairs of $\Sigma_{\boldsymbol{x}}^{1/2} \Sigma_{\boldsymbol{\beta}} \Sigma_{\boldsymbol{x}}^{1/2}$ and $\bar{\tau} = \mathrm{Tr}(\Sigma_{\boldsymbol{x}} \Sigma_{\boldsymbol{\beta}}) + \epsilon^2$. Truncating (55) at data degree three keeps $\sum_j r_j\, \boldsymbol{x}_j^\top \boldsymbol{A}\, \boldsymbol{x}_{L+1}$. The coefficient it assigns to the normalized degree-3 mode is the raw coefficient *times* the mode's norm,

$$v_a^{\rm trunc} = \frac{\tilde{s}_a}{L\tilde{s}_a + \epsilon^2} \sqrt{L(L+1)\tilde{s}_a + L\bar{\tau}}\,, \tag{56}$$

whereas the projection gives the overlap *divided* by the norm (Table 2),

$$v_a^{\rm proj} = \frac{L\,\tilde{s}_a}{\sqrt{L(L+1)\tilde{s}_a + L\bar{\tau}}} = \frac{\tilde{s}_a}{\sqrt{(1+L^{-1})\tilde{s}_a + \bar{\tau}/L}}\,. \tag{57}$$

Their ratio is

$$\frac{v_a^{\rm trunc}}{v_a^{\rm proj}} = \frac{(L+1)\tilde{s}_a + \bar{\tau}}{L\tilde{s}_a + \epsilon^2} = 1 + \frac{\tilde{s}_a + \bar{\tau} - \epsilon^2}{L\tilde{s}_a + \epsilon^2} \;>\; 1, \tag{58}$$

since $\bar{\tau} \geq \epsilon^2$. The truncation therefore always overestimates, for every mode and every parameter setting, and approaches the projection only as $O(L^{-1})$. For example, at $d = 2$, $L = 6$, $\epsilon^2 = 0.5$, $\Sigma_{\boldsymbol{x}} = I_d/2$ and $\Sigma_{\boldsymbol{\beta}} = I_d$, one has $\tilde{s}_a = 0.5$ and $\bar{\tau} = 1.5$, giving a ratio of $5/3.5 \approx 1.43$.

The two operations differ because the eigencoefficients are *projections* of the oracle onto an orthonormal basis, $v_i = \mathbb{E}_Z[y^\star \phi_i]$, while an expansion *truncates* the oracle itself. They agree only when the

discarded terms are orthogonal to the retained modes, and regular polynomials of different degree are not. Here, every higher-order term of (55) generally has a nonzero projection onto the degree-3 mode, so the correct coefficient requires resumming the whole series.

dMMSE. The A mode gives a more extreme case. By Section H.2, the dMMSE contains no A at all, so any expansion of it assigns $A\,\boldsymbol{x}_{L+1}$ a coefficient of zero. The projection is instead $c_b/\sqrt{\mathrm{Var}\,\tau + 2\,\mathbb{E}\tau^2/L}$ with $c = \mathrm{Tr}_1\,\tilde{\kappa}_3 + 2\tilde{\Sigma}\tilde{\mu}$ (Table 2), which is nonzero whenever $c \neq 0$, since A correlates with functions of W and S. For a centred Gaussian prior $c = 0$, so this mode is specific to the memorizing solution.

H.4 Generalizing vs memorizing solution

The dMMSE is a functional of the pool measure, and each of its eigencoefficients depends on finitely many pool moments, which are polynomials in the cumulants. Setting $\hat{\kappa}_{\ell\geq 3} = 0$ throughout is therefore the same as replacing the pool by the Gaussian with the same mean and covariance.

Proposition 2 (The generalizing solution is a cumulant truncation of the memorizing one)**.** *Replacing the pool $\mathcal{B}_{\mathrm{tr}}$ in Equation (54) by $\mathcal{N}(\hat{\mu}_{\boldsymbol{\beta}}, \hat{\Sigma}_{\boldsymbol{\beta}})$ gives*

$$\left(W + \epsilon^2\hat{\Sigma}_{\boldsymbol{\beta}}^{-1}\right)^{-1}\left(S + \epsilon^2\hat{\Sigma}_{\boldsymbol{\beta}}^{-1}\hat{\mu}_{\boldsymbol{\beta}}\right), \tag{59}$$

which is the ridge solution Equation (53) with the prior's mean and covariance replaced by the pool's. When $\hat{\Sigma}_{\boldsymbol{\beta}}$ is singular ($T \leq d$), $\hat{\Sigma}_{\boldsymbol{\beta}}^{-1}$ is infinite off its range, and the predicted task is pinned to $\hat{\mu}_{\boldsymbol{\beta}}$ in directions the pool does not span.

Proof. Under a Gaussian prior and the linear-Gaussian likelihood, the posterior over $\boldsymbol{\beta}$ is Gaussian, and its mean is the displayed expression. □

The two oracles therefore differ in exactly two ways. The higher cumulants $\hat{\kappa}_{\ell\geq 3}$ appear only in the dMMSE. The first two cumulants appear in both, as $(\hat{\mu}_{\boldsymbol{\beta}}, \hat{\Sigma}_{\boldsymbol{\beta}})$ versus $(\boldsymbol{\mu}_{\boldsymbol{\beta}}, \Sigma_{\boldsymbol{\beta}})$. Their difference shrinks as $T^{-1/2}$ entrywise, but is severe for $T \leq d$, where the pool does not span task space.