In-Context Function Learning in Large Language Models

Elif Akata[*12] **Konstantinos Voudouris**[*13] **Vincent Fortuin**[14] **Eric Schulz**[1]
[1] Helmholtz Munich [2] University of Tübingen [3] AI Security Institute [4] University of Technology Nuremberg

Abstract

Large language models (LLMs) can learn from a few demonstrations provided at inference time. We study this in-context learning phenomenon through the lens of Gaussian Processes (GPs). We build controlled experiments where models observe sequences of multivariate scalar-valued function samples drawn from known GP priors. We evaluate prediction error in relation to the number of demonstrations and compare against two principled references: (i) an empirical GP-regression learner that gives a lower bound on achievable error, and (ii) the expected error of a 1-nearest-neighbor (1-NN) rule, which gives a data-driven upper bound. Across model sizes, we find that LLM learning curves are strongly influenced by the function-generating kernels and approach the GP lower bound as the number of demonstrations increases. We then study the inductive biases of these models using a likelihood-based analysis. We find that LLM predictions are most likely under less smooth GP kernels. Finally, we explore whether post-training can shift these inductive biases and improve sample-efficiency on functions sampled from GPs with smoother kernels. We find that both reinforcement learning and supervised fine-tuning can effectively shift inductive biases in the direction of the training data. Together, our framework quantifies the extent to which LLMs behave like GP learners and provides tools for steering their inductive biases for continuous function learning tasks.

[*]Equal contribution.
Correspondence: elif.akata@helmholtz-munich.de

Proceedings of the 29[th] International Conference on Artificial Intelligence and Statistics (AISTATS) 2026, Tangier, Morocco. PMLR: Volume 300.

1 INTRODUCTION

In-context learning (ICL) enables large language models (LLMs) to "learn" a task at inference time by conditioning on a small number of task-relevant input-output pairs (Brown et al., 2020). In-context learning's mechanisms have been probed empirically (Garg et al., 2022; Si et al., 2023) and theoretically (Xie et al., 2021). It has been shown that, surprisingly, transformers can implement linear-regression–style algorithms in context (Akyürek et al., 2022), demonstrations and their labels can be randomized while still achieving improved performance (Min et al., 2022), and more generally, ICL mechanistically resembles gradient-descent dynamics and kernel regression (Von Oswald et al., 2023; Requeima et al., 2024).

While the power of in-context learning for learning continuous functions has been established (Coda-Forno et al., 2023; Requeima et al., 2024), even outperforming some traditional statistical regression techniques (Vacareanu et al., 2024), it remains unclear which functional priors these models have and whether parameter-efficient post-training methods can effectively steer them, thus changing their in-context learning capabilities.

We take a statistical perspective where we:

- cast ICL as non-parametric regression under known Gaussian Process (GP) priors, implementing principled learning-curve evaluations with a GP regression as an empirical lower bound and a derived 1-NN rule as a data-driven upper bound,
- introduce a likelihood-based inductive bias analysis that identifies which GP kernels best explain a model's predictions,
- test whether parameter-efficient post-training methods with supervision and reinforcement learning can steer these implicit priors and improve sample efficiency on deliberately hard kernels.

Empirically, we observe coherent, GP-like behav-

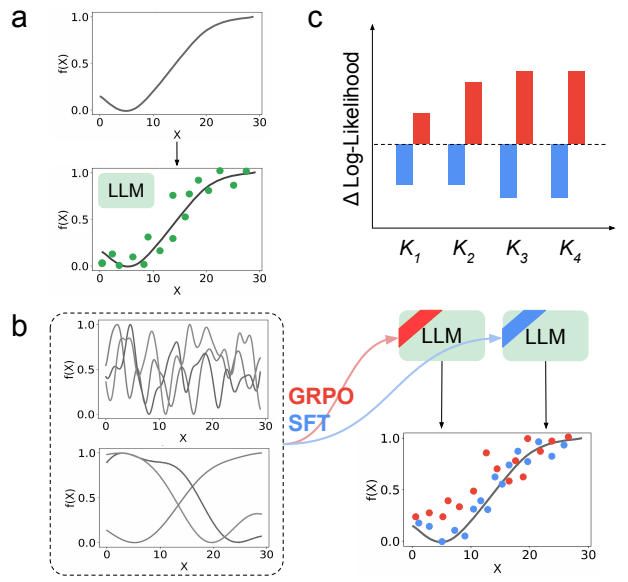

Figure 1: Overview of our framework. (a) In-context function learning in base models: a large language model (LLM) receives demonstrations from functions sampled from known Gaussian-process (GP) priors and predicts $f(\mathbf{X})$ at a new $\mathbf{X}$. (b) Post-training: the model is fine-tuned (SFT or GRPO) and re-evaluated to measure changes in learning curves. (c) Inductive-bias analysis: a likelihood comparison identifies the GP kernels that best explain model predictions before and after training.

ior. Larger LLMs learn faster; smoother kernels give steeper learning curves; and with enough demonstrations, models often approach the empirical GP baseline. Our analyses provide quantitative, interpretable answers to the question of when LLMs act like non-parametric regressors on multivariate function learning tasks. Our inductive bias analysis reveals that LLM predictions are much more likely under rougher, less predictable kernels (with more local and higher variance), such as the Matérn $\frac{1}{2}$ compared to smoother kernels like the Squared Exponential. We find that two parameter-efficient post-training methods are effectively able to shift this inductive bias towards smoother kernels when trained on functions sampled from them, making fine-tuned model predictions much more likely under GPs with the Squared Exponential kernel. Supervised Fine-Tuning (SFT) updates a small fraction of model weights based on labelled data, while reinforcement learning updates these weights by scoring model outputs according to a reward function, here, the error between the LLM's prediction and the true function output. We find some evidence that GRPO produces models with more generalizable in-context learning capabilities, similar to the results of Chu et al. (2025).

Together, these results suggest that even modestly-sized LLMs have impressive in-context learning capabilities. Moreover, post-training methods that update only a fraction of total model weights constitute effective ways to steer inductive biases, with evidence that reinforcement learning is the most generalizable approach. Our experiments showcase how Gaussian Processes offer a principled and tightly controllable testbed for studying in-context learning in LLMs.

2 BACKGROUND

GPs provide a statistical formalism for reasoning about functions and uncertainty. A GP prior defines a distribution over functions via a kernel that encodes smoothness and characteristic length scales; conditioning on observations yields a posterior that achieves Bayes-optimal predictions for the chosen prior. Given data $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^n$ where $\mathbf{x}_i \in \mathbb{R}^d$, prior $f : \mathbb{R}^d \to \mathbb{R} \sim \mathcal{GP}(0, k)$ and noise $y_i = f(\mathbf{x}_i) + \epsilon_i$, $\epsilon_i \sim \mathcal{N}(0, \sigma_\epsilon^2)$, we can define a kernel, $K_\epsilon := k(\mathbf{X}, \mathbf{X}) + \sigma_\epsilon^2 I$ and $k_* := k(\mathbf{X}, \mathbf{x}_*)$. Then

$$p(f_* \mid \mathbf{x}_*, \mathcal{D}) = \mathcal{N}\Big(k_*^\top K_\epsilon^{-1} y, \; k(\mathbf{x}_*, \mathbf{x}_*) - k_*^\top K_\epsilon^{-1} k_*\Big),$$
$$p(y_* \mid \mathbf{x}_*, \mathcal{D}) = \mathcal{N}\Big(k_*^\top K_\epsilon^{-1} y, \; k(\mathbf{x}_*, \mathbf{x}_*) - k_*^\top K_\epsilon^{-1} k_* + \sigma_\epsilon^2\Big).$$

Using this Bayes-optimal reference, we design controlled experiments to test how closely LLMs resemble GP learners. Models receive a list of demonstrations $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^n$ generated from functions drawn from known GP priors (Matérn and Squared Exponential), and must predict y_* at a new $\mathbf{x}_*$. We quantify in-context learning via the absolute-error learning curve

$$\mathcal{E}(n) \;=\; \mathbb{E}[\,|y_* - \hat{y}_*(\mathbf{x}_*; \mathcal{D}_n)|\,]$$

as a function of n. To interpret these curves, we compare against two complementary references evaluated on the same data: (i) GP regression, which serves as a practical lower-bound reference under the specified prior, and (ii) the expected error of a 1-nearest-neighbor rule that returns the value at the closest observed input, providing a data-driven upper bound for memorisation-based strategies.

Beyond aggregate error, our study explores the inductive biases that guide a model's predictions. Thus, we propose a likelihood-based inductive bias analysis: given a history of demonstrations, we compute

the GP posterior predictive distribution under different kernel families and record the log-likelihood of the model's prediction. Aggregating these scores across tasks and number of demonstrations identifies which kernels best explain the model's behavior, resulting in a data-driven summary of its implicit inductive bias. Finally, we examine whether these inductive biases can be changed through post-training. We study supervised fine-tuning (SFT) and reinforcement learning (with group-relative policy optimization, GRPO) with training curricula targeted at the kernels under which the base model is least consistent, and evaluate resulting changes in both priors and learning curves.

3 METHODS

3.1 Experiment 1: In-Context Learning Curves

We first study the LLM generalization error as a function of the number of demonstrations, on functions generated from different kernels, comparing them to two reference baselines described below.

Large Language Models In the main paper, we evaluate models from the Qwen-3 model family (Yang et al., 2025), using the 8-billion, 14-billion, and 32-billion parameter versions with 4-bit bits-and-bytes quantization to increase inference speed (Dettmers et al., 2022a,b, 2023). As comparisons, in Section 7 of the supplementary material, we also evaluate three members of the Llama family (Llama-3.2-3B-Instruct, Llama-3.1-8B-Instruct, Llama-3.1-70B-Instruct; Grattafiori et al., 2024), three members of the Gemma-3 family (4B, 12B, and 27B; Gemma Team et al., 2024), and two members of the Mistral family (Mistral-7B-Instruct-v0.3 and Mistral-Small-24B-Instruct-2501; Jiang et al., 2023), where B denotes the parameter size in billions and all models are 4-bit quantized.

Evaluation Data & Procedure We construct functions $f : \mathbb{R}^d \to \mathbb{R}$ for $d \in \{1, 2, 3, 4\}$. For each dimensionality, we sample 200 functions from four kernels, the Matérn $\frac{1}{2}$, $k_m(\delta)$ and the Squared Exponential, $k_s(\delta)$, with length scales, $\ell \in \{1, 8\}$:

$$k_m(\delta) = \sigma_f^2 \frac{2^{1-\nu}}{\Gamma(\nu)} \left(\frac{\sqrt{2\nu}\,\delta}{\ell} \right)^{\nu} K_\alpha \left(\frac{\sqrt{2\nu}\,\delta}{\ell} \right)$$

$$k_s(\delta) = \sigma_f^2 \exp\left(-\frac{\delta^2}{2\ell} \right)$$

where δ is the Euclidean distance between two values, x, x', σ_f^2 is the output variance, ν is the *smoothness* (set to $\nu = \frac{1}{2}$ for both Matérn kernels), $\Gamma(\cdot)$ is the gamma function, and $K_\alpha(\cdot)$ is the modified Bessel function of the second kind with order $\alpha = \nu + \frac{1}{2}$. We use these kernels because they differ maximally in smoothness: the Matérn $\frac{1}{2}$, $k_m(\delta)$ is differentiable nowhere while the Squared Exponential is infinitely differentiable, being $\lim_{\nu\to\infty} k_m(\delta)$. We use two values of ℓ to vary the scale over which the functions are correlated, again varying their predictability and smoothness. We use $\sigma_f^2 = 0.001$ for all kernels.

From each function, we compute outputs, y, for 50 randomly sampled inputs over an arbitrary range, $x \sim \mathcal{U}[0, 29]$. We add Gaussian distributed noise, $\epsilon \sim \mathcal{N}(0, \sigma_\epsilon^2)$ with $\sigma_\epsilon^2 = 0.001$. The LLMs are prompted with the text described in the supplementary material (Section 3), including between 0 and 49 demonstrations of $(\mathbf{x}, y)$ pairs. For a new $\mathbf{x}$, we compute the absolute error between the LLM's prediction, $\hat{y}$, and the true value (plus noise), $\tilde{y}$. For each n demonstrations, we compute the mean absolute error across all functions.

Baselines We use two reference baselines to which we compare the LLM learning curves. The first is the empirical error from a GP regression trained on the same data as the LLMs, generated from functions drawn from its kernel. The second is the expected error for a k-Nearest Neighbor (k-NN) algorithm, with $k = 1$. The GP baseline produces an empirical lower bound on the error after n samples. The k-NN baseline produces a reasonable expected upper bound on the error for the case where the LLM simply returns the y-value for the numerically closest x-value it has seen so far.

To compute the 1-NN expected absolute error, we assume n inputs are drawn uniformly as $X \sim \mathcal{U}[0, L]$ with some upper bound, L (here, $L = 29$). We approximate the expected absolute error by numerically evaluating the integral:

$$\mathbb{E}[|\varepsilon|] = \sqrt{\frac{2}{\pi}} \int_0^{L/2} V(d)\, \frac{2(n-1)}{L} \left(1 - \frac{2d}{L} \right)^{n-2} \mathrm{d}d$$

where $0 \le d \le \frac{L}{2}$, and $V(d) = 2\sigma_f^2 + 2\sigma_\epsilon^2 - 2k(d)$ where $k(\cdot)$ is the GP kernel function with output variance σ_f^2. Further information on deriving this expectation is included in the supplementary material (Section 2).

3.2 Experiment 2: Inductive Bias Analysis

To infer the expectations of the LLMs towards learning functions, we borrow a methodology from cognitive science (Griffiths et al., 2008; Li et al.,

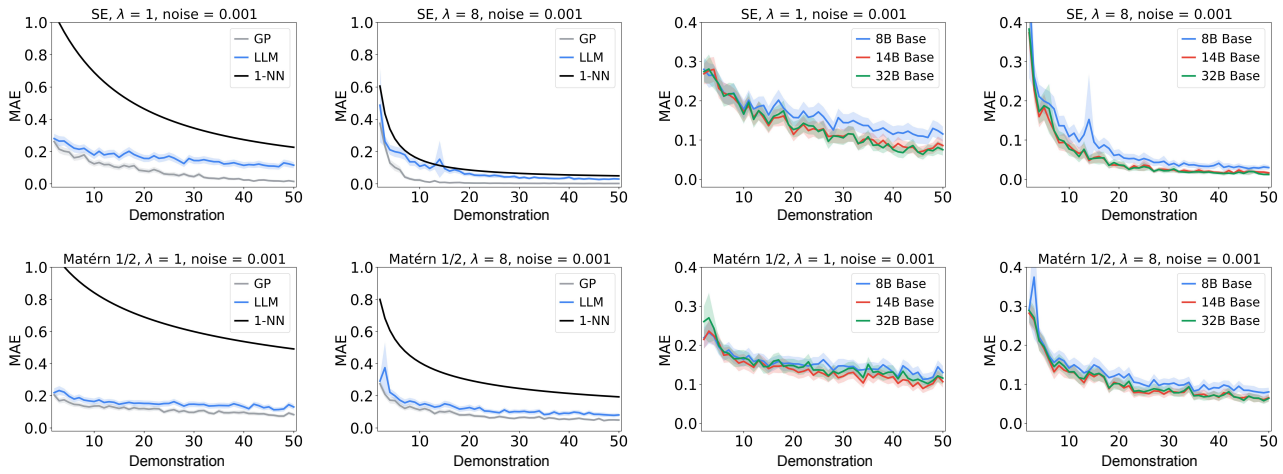

Figure 2: Learning curve analysis on 1-dimensional functions. The mean absolute error after n demonstrations by function type. Left Four: Qwen-3-8B learning curves four functions drawn from four kernels, compared to the error of a GP regression and the expected error of a 1-nearest neighbor rule. The LLM learning curves generally approach the GP regression baseline and are well below the 1-NN rule. Right Four: Model size comparisons between the 8B, 14B, and 32B Qwen-3 models, on identical data. The 14B and 32B models show noticeably lower error rates, but do not differ significantly from each other, suggesting a logarithmic scaling law. All LLM and GP learning curves are shown with 95% bootstrapped confidence intervals.

2023; Lucas et al., 2015; Schulz et al., 2018; Wilson et al., 2015). For each x_n, we infer the posterior probability distribution over Y for some GP given $\{(\mathbf{x}_1, y_1), ..., (\mathbf{x}_{n-1}, y_{n-1})\}$. We then compute the log-likelihood of the LLM's point prediction given the same data. Subsequently, we compare the mean likelihood over all predictions, under GPs with different kernels. This measures how likely the LLM responses are assuming different kernels. We use GPs with fixed kernel variances (1), length scales (8), and the same noise variance as σ_ϵ^2 (0.001). We compare the Matérn kernels with $\nu \in \{0.5, 1.5, 2.5\}$ with length scales $\ell \in \{1, 8\}$ and the Squared Exponential kernel with length scales $\ell \in \{1, 2, 3, 4, 5, 6, 7, 8\}$. We hypothesize that the most likely kernel approximates the inductive bias of the model.

3.3 Experiment 3: Post-Training

The inductive bias analysis is used to reveal what priors LLMs have about functions. We can use this information to examine whether common parameter-efficient post-training methods differ in their ability to shift those expectations. We explore two approaches to parameter-efficient fine-tuning with low-rank adapters: supervised fine-tuning and reinforcement learning.

Parameter-efficient fine-tuning with QLoRA We use Quantized Low Rank Adaptation (QLoRA) to efficiently and scalably fine-tune large language models by inserting small, low-rank matrices layer-wise and updating only these weights and freezing the rest (Dettmers et al., 2023). This means that gradients are computed and weights are updated for only a fraction of the total parameters in the model. Specifically, for the weight matrix, W, of a transformer layer, we inject an adapter matrix, W_a, which is the product of two low-rank matrices, $L_1 \in \mathbb{R}^{d \times r}$ and $L_2 \in \mathbb{R}^{r \times k}$, where d, k are the input and output dimensionalities respectively and $r \ll d, k$. During a forward pass, inputs of length d are passed to W and W_a independently, and outputs of length k are summed, subject to some scaling factor $\frac{r}{\alpha}$. We choose standard values of $r = \alpha = 16$, such that outputs from W and W_a are weighted equally. We inject adapters in all layers. We also make use of quantized models, where model and adapter weights are dynamically set to lower precision, to speed up training.

Dataset We construct a training dataset informed by the analyses conducted in Experiment 2. We choose a kernel under which the base models' predictions are least likely, and sample functions and in-context learning data for each, matching the test dataset in structure.

Supervised Fine-Tuning (SFT) We update the weights of the low-rank adapters using this supervised dataset. We update model weights using batch gradient descent with the token-level cross-entropy loss:

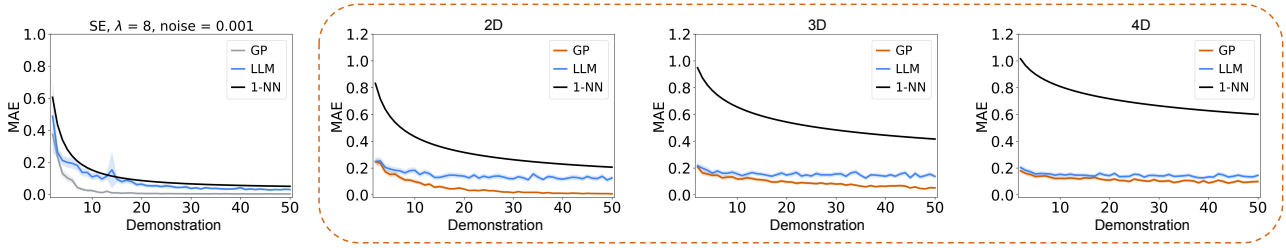

Figure 3: Qwen-3-14B learning curve comparison for 1-, 2-, 3-, 4-dimensional data drawn from the Squared Exponential with $\lambda = 8$. The mean absolute error after n demonstrations by function type. All LLM and GP learning curves are shown with 95% bootstrapped confidence intervals.

$$\mathcal{L}(\theta) = -\sum_{t=1}^{T} \log\, p_\theta(y_t | y_{<t})$$

where θ is the set of adapter weights, T is the size of the ordered set of target completion tokens given a prompt, y_t is the target token at step t, and $y_{<t}$ is the set of ordered target completion tokens prior to t.

Reinforcement Learning (RL) We use the online RL algorithm *Group Relative Policy Optimisation* (GRPO) to update the adapter weights. In the RL setting, the set of all model and adapter weights can be considered the *policy*, π_θ, which takes textual inputs (observations) and produces a token (actions). For each batch of M prompts, $\{q_1, ..., q_M\}$ in the dataset, the model produces a set of N completions, $\{o_1, ..., o_N\}$. These completions are assigned a reward using a reward model, giving a set of rewards $\{r_1, ..., r_N\}$.

We compute the loss for some prompt q as follows:

$$\mathcal{L}(\theta) = -\frac{1}{\sum_{i=1}^{N} |o_i|} \sum_{i=1}^{N} \sum_{t=1}^{|o_i|} \left[\min\left(\frac{\pi_\theta(o_{i,t} \mid q, o_{i,<t})}{\pi_{\theta_{\text{old}}}(o_{i,t} \mid q, o_{i,<t})} \hat{A}_{i,t}, \right. \right.$$
$$\left. \left. \text{clip}_\eta\left(\frac{\pi_\theta(o_{i,t} \mid q, o_{i,<t})}{\pi_{\theta_{\text{old}}}(o_{i,t} \mid q, o_{i,<t})} \right) \hat{A}_{i,t} \right) \right]$$

where $|o_i|$ is the length, in tokens, of o_i, clip$(\cdot)$ clips its argument between $1 \pm \eta$, and $\hat{A}_{i,t}$ is the *advantage*—the normalized reward for output o_i:

$$\hat{A}_{i,t} = \frac{r_i - \text{mean}(\{r_1, ..., r_n\})}{\text{std}(\{r_1, ..., r_n\})}$$

Following common practice, we exclude the usual KL-divergence term when computing the loss (Hu et al., 2025; Liu et al., 2025; Yu et al., 2025). We update the adapter weights using gradient ascent over this loss function.

Reward Function Models trained with GRPO are rewarded using the negative absolute difference between the last float in their completion and the true $\tilde{y}$ (including additive noise). This reward is capped at a minimum of -10 for parseable responses ($10\times$ the range of possible y-values), and is -11 for any completions that do not contain a parseable float. As an alternative, we also conduct parallel training with a log-likelihood-based reward function. Here, the reward is the log likelihood of the model's response under the GP identified in Experiment 2. We cap the reward arbitrarily at a minimum of -999, and set the reward for unparseable responses at -1000. We find no interpretative differences between these two reward functions.

An overview of the general, model-agnostic procedure we took for analysing inductive biases in LLMs on continuous function learning tasks can be found in supplementary material (Section 1). All models were fine-tuned and evaluated on 80 GB NVIDIA A100 GPUs on our SLURM-based internal compute cluster (HMGU HPC), using approximately 80 GPU-hours in total.

4 RESULTS

4.1 Experiment 1: In-Context Learning Curves

First, we confirm that LLMs are capable of doing in context regression with novel multivariate scalar-valued functions (Requeima et al., 2024; Si et al., 2023; Vacareanu et al., 2024), with errors that are generally close to empirical GP regression and well below the error of a 1-Nearest Neighbor algorithm (Figures 2 and 3). Indeed, for functions sampled from kernels with low length scales, LLM error is lower than the GP regression given a few demonstrations. This suggests that these models are not simply recalling information from earlier in their context window, but that they are able to fit complex functions in their attention matrices. It is particularly impressive that these models are able to sample-efficiently and accurately predict scalar

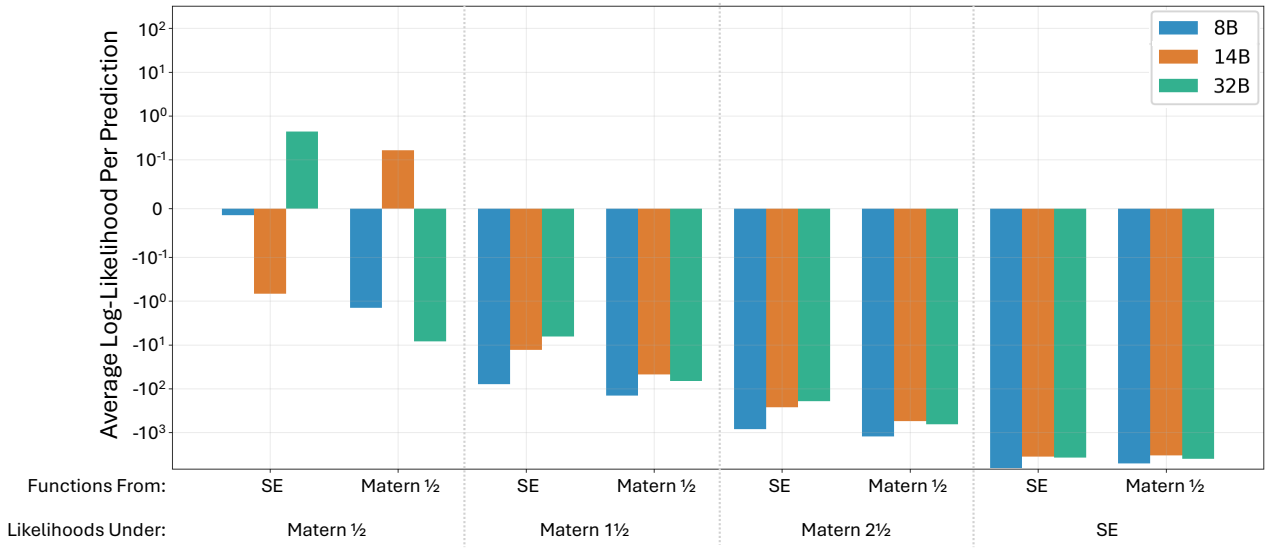

Figure 4: Inductive bias analysis of the base models (8B, 14B, 32B). The average likelihood per prediction is computed under GPs with four different kernels ($\ell = 8$), on 1-dimensional data drawn from either the Squared Exponential or the Matérn $\frac{1}{2}$. These likelihoods are presented on a symmetric log scale. LLM predictions for all model sizes are more likely under kernels with lower ν, i.e., those that describe rougher, less predictable functions.

outputs given 4-dimensional input. Indeed, we find no noticeable difference between the error rates on lower-dimensional and higher-dimensional function learning tasks.

We find an appreciable performance boost between the 8B and larger models, but that difference is not significant between 14B and 32B models. This is indicative of a logarithmic scaling law when it comes to in-context function learning. However, further research with more granular model size differences would be required to confirm this conclusion.

4.2 Experiment 2: Inductive Bias Analysis

We then examined how likely the LLMs' predictions were under GPs with different kernels. Figure 4 presents the average likelihood of the LLMs' predictions under four different kernels, on functions sampled from either the Squared Exponential (SE; smooth functions) or the Matérn $\frac{1}{2}$. We find that, for all model sizes and function families (SE, Matérn $\frac{1}{2}$), the LLMs' predictions are most likely under the Matérn $\frac{1}{2}$ and least likely under the SE for 1-dimensional functions. Indeed, we see a general trend that as $\nu \to \infty$, LLM predictions become less likely. We found the same pattern applies to models from other families, such as Gemma and Llama, as shown in the supplementary material (Section 7). Within each kernel, we also varied the length scale. We found that LLM predictions are more likely under lower values of λ compared to higher values, as shown in the supplementary material (Section 5).

It is also the case that, in general, predictions are more likely under rougher kernels because the posterior over predictions is generally broader. Indeed, we verified this by conducting an identical analysis in which independent GPs are tasked with making predictions given some data, and then examined under which kernels their predictions were most likely. We found the same qualitative trend as with the LLMs—the predictions of an SE GP on SE sampled functions were much more likely under the Matérn $\frac{1}{2}$ than under the SE. We outline one strategy for adjusting for this *variance inflation effect* in the supplementary material (Section 8). Applying this strategy makes the LLM's predictions more likely under smoother kernels in some cases. Nonetheless, even after this correction, LLM predictions are more likely under shorter length scales (with the Squared Exponential kernel), indicating inductive biases for less predictable functions.

In contrast, we found that LLM predictions became more likely under smoother kernels as the dimensionality of the function input increased. While the kernel with the highest log-likelihood for 1-dimensional functions is the Matérn $\frac{1}{2}$ (-2.59×10^2), it is the Matérn $1\frac{1}{2}$ for 2-dimensional functions (-2.05×10^3), and the Squared Exponential for 3- and 4-dimensional functions (-6.92×10^3 and -1.43×10^4 respectively). This indicates that the model's implicit prior adapts to-

wards smoother kernels in higher-dimensional spaces and suggests that these these LLMs favour higher global regularity when modelling functions in these spaces.

4.3 Experiment 3: Post-Training

We fine-tuned models on one-dimensional functions sampled from the squared exponential kernel with $\lambda = 8$, the kernel under which our LLM predictions were least likely. Figure 5 shows the differences in likelihoods between the base model and the fine-tuned model for four fine-tuning checkpoints (1,000, 2,000, 5,000, 10,000 steps). In general, the likelihood of LLM predictions increases by a greater degree as the kernels become less smooth (i.e., $\nu \to \infty$). However, we see that the SFT-trained model predictions tend to be less likely than the base model's when the functions are drawn from the Matérn $\frac{1}{2}$. This suggests that SFT may be overfitting to the functional form of functions drawn from the Squared Exponential kernel on which they were trained. In contrast, the GRPO-trained models see comparable likelihood boosts for both data regimes. Together, these results cohere with existing findings which suggest that while SFT memorizes training data, reinforcement learning-based training can contribute to generalizing over that training data (Chu et al., 2025).

Figure 6 shows the learning curves of models on novel held-out test data. We find that both SFT and GRPO push the absolute error for all n down towards the empirical GP baseline, indicating that post-training only a fraction of the total model parameters is a powerful method for altering in-context learning capabilities. Further results from training on different data, using different reward functions and data in higher dimensions are presented in the supplementary material (Sections 4 & 6).

5 DISCUSSION

Function learning is a powerful test bed for measuring the ability of models to generalize to superficially novel yet functionally equivalent in-context learning problems. By exploiting the principled statistical framework of Gaussian Process regression, we can further characterize the priors that LLMs have about continuous functions, and the power of post-training methods to steer those inductive biases. Indeed, a key debate in natural language processing is the differential power of token-based reinforcement learning compared to supervised fine-tuning (Chu et al., 2025).

In this study, we found evidence of strong in-context learning capabilities for novel multivariate function learning problems, confirming existing results (Nafar et al., 2024; Requeima et al., 2024; Vacareanu et al., 2024). The error rates on these problems are surprisingly low, competitive with the empirical GP and far outperforming a data-driven 1-nearest-neighbor algorithm, even for higher-dimensional problems. This suggests that these models are not simply deriving predictions from previously observed demonstrations (Vacareanu et al., 2024).

We also found that LLMs appear to be inductively biased towards rougher continuous functions for lower dimensional problems and smoother continuous functions for higher dimensional problems. This is counter-intuitive in the context of function learning in humans, who appear to show a bias towards smoother, more predictable functions in all regimes (Schulz et al., 2015). The exact reason for this phenomenon may either derive from features of the attention mechanism (Xie et al., 2021) or from properties of the large-scale pre-training data used to train these models (Cruz et al., 2023).

Both supervised fine-tuning and group-relative policy optimization over a small set of injected low-rank adapter matrices on each layer where able to effectively shift these preferences towards the structure of the training data. Furthermore, we found some evidence pointing in the same direction as Chu et al. (2025), suggesting that reinforcement-based post-training (e.g., GRPO) leads to more generalizable behavior than supervised fine-tuning, which tends to memorize the training data.

There are several limitations to this work. First, our inductive bias analysis suffers from a variance inflation problem, meaning that model predictions are always as or more likely under rougher kernels due to larger variances. Future work will explore alternative corrections for this beyond the case described in the supplementary material (Section 8). Second, it is not yet clear how our results generalize from the controlled setting of function learning to more ecologically valid in-context learning settings, such as interacting with a user or iterating over a novel coding task.

6 RELATED WORK

In-Context Learning (ICL) in Large Language Models was most prominently discussed by Brown et al. (2020), who showed that large-scale autoregressive models can learn to solve a wide range of tasks purely from demonstrations at inference time. Subsequent work has sought to reveal the mechanisms underlying this capacity. Xie et al. (2021) interpret ICL as implicit Bayesian inference over latent concepts, while Akyürek et al. (2022) formulate it as an approxima-

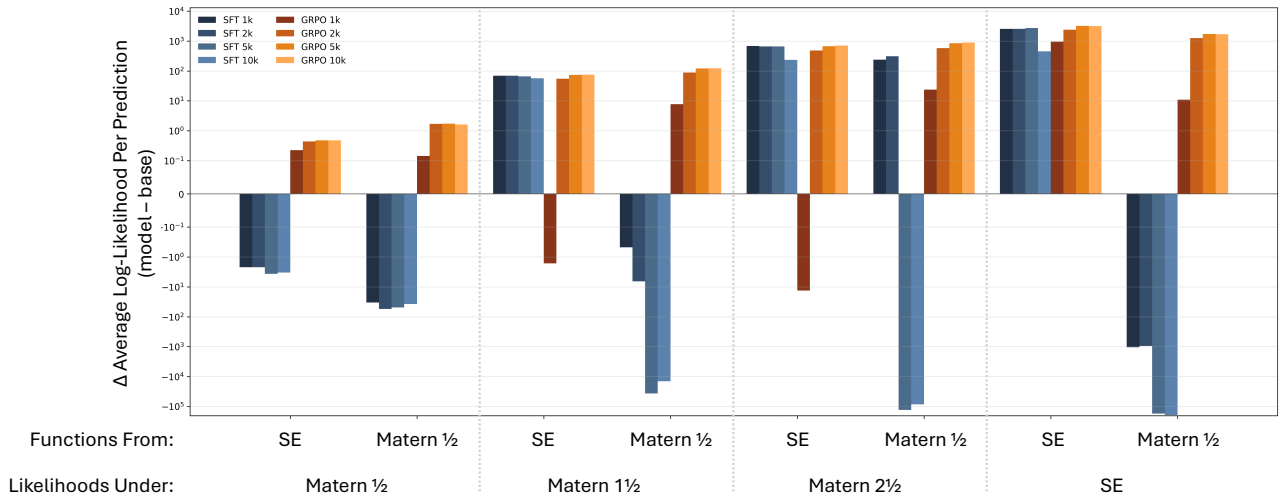

Figure 5: Inductive bias analysis compared to the base model for the 8B model for three checkpoints (1k, 2k, 5k, 10k steps). Average likelihood per prediction shown on a symlog scale.

tion of gradient descent performed during the forward pass (Von Oswald et al., 2023). Alternative perspectives include structure induction, where transformers recombine latent compositional patterns learned during pre-training to induce task structure (Hahn and Goyal, 2023). Empirically, larger models appear to display qualitatively different sensitivities to context noise compared to smaller models (Wei et al., 2023), and prior work has examined whether models can learn specific function classes such as linear and sparse regressions and how they compare to classical estimators (Bhattamishra et al., 2023; Garg et al., 2022).

Inductive Biases in ICL Wei et al. (2023) investigated feature biases using underspecified prompts, showing that GPT models systematically prefer certain predictive features (e.g., sentiment over lexical markers). Coda-Forno et al. (2023) find that GPT-3 exhibits biases in favour of positive monotonic functions, but that these biases can be shifted through in-context demonstrations. This connects to a long tradition in computational cognitive science, where inductive biases are inferred from point estimates in function learning tasks (Griffiths et al., 2008; Li et al., 2023; Lucas et al., 2015; Schulz et al., 2018; Wilson et al., 2015). Humans themselves appear to favour smoother, more predictable functions (Schulz et al., 2015).

Regression and Function Learning with LLMs Recent work has examined regression capabilities of LLMs directly. Vacareanu et al. (2024) show that models such as GPT-4 and Claude 3 can outperform classical supervised learners on certain non-linear regression tasks, while Nafar et al. (2024) demonstrate that performance depends strongly on the presentation of in-context demonstrations. To better characterise predictive behaviour, Requeima et al. (2024) introduce LLM Processes, eliciting predictive distributions directly from LLMs and comparing them against regression baselines. Several theoretical studies further suggest that transformers may implicitly implement kernel or least-squares regression mechanisms (Han et al., 2023; Zhang et al., 2024; Sun et al., 2025; Bai et al., 2023). Our work builds on these insights by evaluating LLMs on controlled function learning tasks and testing how well Gaussian Process (GP) models with appropriate kernels can reproduce their in-context predictions.

Bayesian Perspectives, Calibration, and Uncertainty From a Bayesian viewpoint, Zhang et al. (2023) argue that ICL corresponds to approximate Bayesian model averaging over latent functions implemented through attention, while Falck et al. (2024) show that LLMs violate certain Bayesian properties such as the martingale condition. Related work studies reliability and uncertainty estimation: Chen and Li (2023) introduce Sparse Gaussian Process Attention to improve calibration, and Jesson et al. (2024) quantify hallucination rates as low-likelihood outputs under an assumed latent model. Rather than modifying model architecture, our approach analyses the natural calibration of LLM outputs on function learning tasks and evaluates how well GP post-hoc modelling captures predictive uncertainty.

Post-Training and Steering Inductive Biases Several studies examine how post-training methods can shift inductive biases. Reinforcement learning from human feedback appears to bias models toward

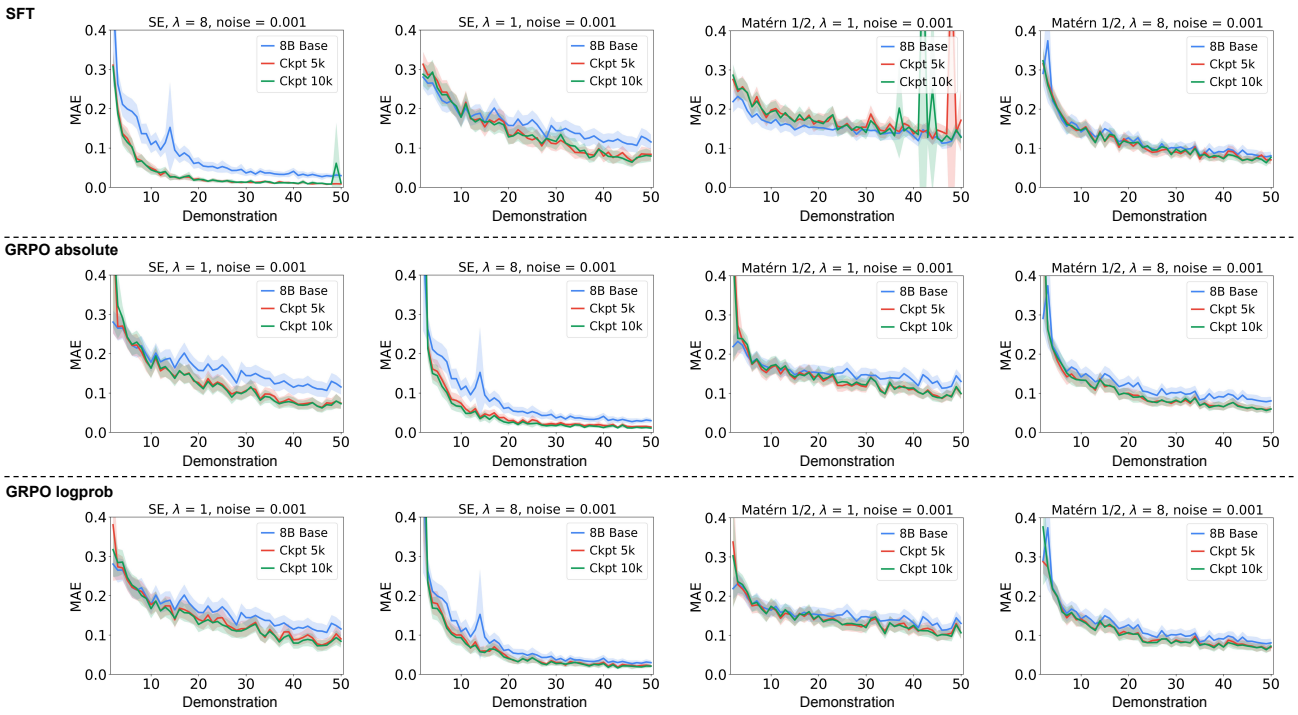

Figure 6: Post training learning curves. Top row: Effects of Supervised Fine-Tuning. 8B base model compared to two checkpoints during training (5k, 10k steps). Middle row: GRPO progression. 8B base model compared to two checkpoints (5k, 10k steps). Bottom row: GRPO progression with a log-likelihood-based reward function. All empirical learning curves are shown with 95% bootstrapped confidence intervals.

extractable features (Cruz et al., 2023), and reinforcement learning-based post-training may improve generalisation compared to supervised fine-tuning (Chu et al., 2025). Our work complements these findings by studying how fine-tuning influences inductive biases in controlled function learning settings.

7 CONCLUSION

Any system interacting with an environment inevitably handles continuous functional relationships, from judging the sentiment of a piece of text to forecasting the weather. Although LLMs are often evaluated on discrete tasks, their routine use often requires learning smooth input-output relationships like mapping text to scores, rewards, or making continuous predictions from a few examples.

Our study casts in-context function learning as a principled test bed for probing the statistical capabilities and inductive biases of large language models. By grounding our analysis in Gaussian process regression, we demonstrate that LLMs can approximate non-parametric regression learners, with performance scaling with model size and approaching the GP lower bound with more demonstrations. However, our inductive bias analysis reveals a consistent tendency toward rougher functions, pointing to a divergence from human expectations.

We further show that post-training with parameter-efficient methods such as supervised fine-tuning and reinforcement learning can shift these biases towards previously unlikely kernels. While supervised fine-tuning exhibits signs of overfitting, reinforcement learning appears to promote more generalizable adjustments. These findings highlight both the promise and limitations of LLMs as function learners, and point to controlled function learning tasks as a powerful framework for dissecting and steering in-context learning mechanisms.

Acknowledgements

This project has received funding from the European Research Council (ERC) under the European Union's Horizon Europe research and innovation programme (ERC Starting Grant TACOS). We thank the International Max Planck Research School for Intelligent Systems (IMPRS-IS) for supporting EA. VF was supported by the Branco Weiss Fellowship.

References

Ekin Akyürek, Dale Schuurmans, Jacob Andreas, Tengyu Ma, and Denny Zhou. What learning algorithm is in-context learning? investigations with linear models. *arXiv preprint arXiv:2211.15661*, 2022.

Yu Bai, Fan Chen, Huan Wang, Caiming Xiong, and Song Mei. Transformers as statisticians: Provable in-context learning with in-context algorithm selection. *Advances in neural information processing systems*, 36:57125–57211, 2023.

Satwik Bhattamishra, Arkil Patel, Phil Blunsom, and Varun Kanade. Understanding in-context learning in transformers and llms by learning to learn discrete functions. *arXiv preprint arXiv:2310.03016*, 2023.

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.

Wenlong Chen and Yingzhen Li. Calibrating transformers via sparse gaussian processes. *arXiv preprint arXiv:2303.02444*, 2023.

Tianzhe Chu, Yuexiang Zhai, Jihan Yang, Shengbang Tong, Saining Xie, Dale Schuurmans, Quoc V Le, Sergey Levine, and Yi Ma. Sft memorizes, rl generalizes: A comparative study of foundation model post-training. *arXiv preprint arXiv:2501.17161*, 2025.

Julian Coda-Forno, Marcel Binz, Zeynep Akata, Matt Botvinick, Jane Wang, and Eric Schulz. Meta-in-context learning in large language models. *Advances in Neural Information Processing Systems*, 36:65189–65201, 2023.

Diogo Cruz, Edoardo Pona, Alex Holness-Tofts, Elias Schmied, Víctor Abia Alonso, Charlie Griffin, and Bogdan-Ionut Cirstea. Reinforcement learning fine-tuning of language models is biased towards more extractable features. *arXiv preprint arXiv:2311.04046*, 2023.

Tim Dettmers, Mike Lewis, Younes Belkada, and Luke Zettlemoyer. Llm.int8(): 8-bit matrix multiplication for transformers at scale. *arXiv preprint arXiv:2208.07339*, 2022a.

Tim Dettmers, Mike Lewis, Sam Shleifer, and Luke Zettlemoyer. 8-bit optimizers via block-wise quantization. *9th International Conference on Learning Representations, ICLR*, 2022b.

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. Qlora: Efficient finetuning of quantized llms. *arXiv preprint arXiv:2305.14314*, 2023.

Fabian Falck, Ziyu Wang, and Chris Holmes. Is in-context learning in large language models bayesian? a martingale perspective. *arXiv preprint arXiv:2406.00793*, 2024.

Shivam Garg, Dimitris Tsipras, Percy S Liang, and Gregory Valiant. What can transformers learn in-context? a case study of simple function classes. *Advances in neural information processing systems*, 35:30583–30598, 2022.

Gemma Team, Thomas Mesnard, Cassidy Hardin, Robert Dadashi, Surya Bhupatiraju, Shreya Pathak, Laurent Sifre, Morgane Rivière, Mihir Sanjay Kale, Juliette Love, et al. Gemma: Open models based on gemini research and technology. *arXiv preprint arXiv:2403.08295*, 2024.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

Thomas L Griffiths, Chris Lucas, Joseph Williams, and Michael Kalish. Modeling human function learning with gaussian processes. *Advances in neural information processing systems*, 21, 2008.

Michael Hahn and Navin Goyal. A theory of emergent in-context learning as implicit structure induction. *arXiv preprint arXiv:2303.07971*, 2023.

Chi Han, Ziqi Wang, Han Zhao, and Heng Ji. Explaining emergent in-context learning as kernel regression. *arXiv preprint arXiv:2305.12766*, 2023.

Jingcheng Hu, Yinmin Zhang, Qi Han, Daxin Jiang, Xiangyu Zhang, and Heung-Yeung Shum. Open-reasoner-zero: An open source approach to scaling up reinforcement learning on the base model. *arXiv preprint arXiv:2503.24290*, 2025.

Andrew Jesson, Nicolas Beltran-Velez, Quentin Chu, Sweta Karlekar, Jannik Kossen, Yarin Gal, John P Cunningham, and David Blei. Estimating the hallucination rate of generative ai. *Advances in Neural Information Processing Systems*, 37:31154–31201, 2024.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. Mistral 7b, 2023. URL `https://arxiv.org/abs/2310.06825`.

Michael Y Li, Fred Callaway, William D Thompson, Ryan P Adams, and Thomas L Griffiths. Learning

to learn functions. *Cognitive science*, 47(4):e13262, 2023.

Zichen Liu, Changyu Chen, Wenjun Li, Penghui Qi, Tianyu Pang, Chao Du, Wee Sun Lee, and Min Lin. Understanding r1-zero-like training: A critical perspective. *arXiv preprint arXiv:2503.20783*, 2025.

Christopher G Lucas, Thomas L Griffiths, Joseph J Williams, and Michael L Kalish. A rational model of function learning. *Psychonomic bulletin & review*, 22(5):1193–1215, 2015.

Sewon Min, Xinxi Lyu, Ari Holtzman, Mikel Artetxe, Mike Lewis, Hannaneh Hajishirzi, and Luke Zettlemoyer. Rethinking the role of demonstrations: What makes in-context learning work?, 2022. URL `https://arxiv.org/abs/2202.12837`.

Aliakbar Nafar, Kristen Brent Venable, and Parisa Kordjamshidi. Learning vs retrieval: The role of in-context examples in regression with llms. *arXiv e-prints*, pages arXiv–2409, 2024.

James Requeima, John Bronskill, Dami Choi, Richard E Turner, and David Duvenaud. Llm processes: Numerical predictive distributions conditioned on natural language. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang, editors, *Advances in Neural Information Processing Systems*, volume 37, pages 109609–109671. Curran Associates, Inc., 2024.

Eric Schulz, Joshua B Tenenbaum, David N Reshef, Maarten Speekenbrink, and Samuel J Gershman. Assessing the perceived predictability of functions. In *Proceedings of the Annual Meeting of the Cognitive Science Society*, volume 37, 2015.

Eric Schulz, Maarten Speekenbrink, and Andreas Krause. A tutorial on gaussian process regression: Modelling, exploring, and exploiting functions. *Journal of mathematical psychology*, 85:1–16, 2018.

Chenglei Si, Dan Friedman, Nitish Joshi, Shi Feng, Danqi Chen, and He He. Measuring inductive biases of in-context learning with underspecified demonstrations. *arXiv preprint arXiv:2305.13299*, 2023.

Haoyuan Sun, Ali Jadbabaie, and Navid Azizan. In-context learning of polynomial kernel regression in transformers with glu layers. *arXiv e-prints*, pages arXiv–2501, 2025.

Robert Vacareanu, Vlad-Andrei Negru, Vasile Suciu, and Mihai Surdeanu. From words to numbers: Your large language model is secretly a capable regressor when given in-context examples. *arXiv preprint arXiv:2404.07544*, 2024.

Johannes Von Oswald, Eyvind Niklasson, Ettore Randazzo, João Sacramento, Alexander Mordvintsev, Andrey Zhmoginov, and Max Vladymyrov. Transformers learn in-context by gradient descent. In *International Conference on Machine Learning*, pages 35151–35174. PMLR, 2023.

Jerry Wei, Jason Wei, Yi Tay, Dustin Tran, Albert Webson, Yifeng Lu, Xinyun Chen, Hanxiao Liu, Da Huang, Denny Zhou, et al. Larger language models do in-context learning differently. *arXiv preprint arXiv:2303.03846*, 2023.

Andrew G Wilson, Christoph Dann, Chris Lucas, and Eric P Xing. The human kernel. *Advances in neural information processing systems*, 28, 2015.

Sang Michael Xie, Aditi Raghunathan, Percy Liang, and Tengyu Ma. An explanation of in-context learning as implicit bayesian inference. *arXiv preprint arXiv:2111.02080*, 2021.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Weinan Dai, Tiantian Fan, Gaohong Liu, Lingjun Liu, et al. Dapo: An open-source llm reinforcement learning system at scale. *arXiv preprint arXiv:2503.14476*, 2025.

Ruiqi Zhang, Spencer Frei, and Peter L Bartlett. Trained transformers learn linear models in-context. *Journal of Machine Learning Research*, 25(49):1–55, 2024.

Yufeng Zhang, Fengzhuo Zhang, Zhuoran Yang, and Zhaoran Wang. What and how does in-context learning learn? bayesian model averaging, parameterization, and generalization, 2023. *URL https://arxiv. org/abs/2305.19420*, 2023.

Checklist

1. For all models and algorithms presented, check if you include:
   (a) A clear description of the mathematical setting, assumptions, algorithm, and/or model. [Yes]
   (b) An analysis of the properties and complexity (time, space, sample size) of any algorithm. [Yes]
   (c) (Optional) Anonymized source code, with specification of all dependencies, including external libraries. [Yes]

2. For any theoretical claim, check if you include:
   (a) Statements of the full set of assumptions of all theoretical results. [Not Applicable]
   (b) Complete proofs of all theoretical results. [Not Applicable]
   (c) Clear explanations of any assumptions. [Not Applicable]

3. For all figures and tables that present empirical results, check if you include:
   (a) The code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL). [Yes]
   (b) All the training details (e.g., data splits, hyperparameters, how they were chosen). [Yes]
   (c) A clear definition of the specific measure or statistics and error bars (e.g., with respect to the random seed after running experiments multiple times). [Yes]
   (d) A description of the computing infrastructure used. (e.g., type of GPUs, internal cluster, or cloud provider). [Yes]

4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets, check if you include:
   (a) Citations of the creator If your work uses existing assets. [Yes]
   (b) The license information of the assets, if applicable. [Not Applicable]
   (c) New assets either in the supplemental material or as a URL, if applicable. [Yes]
   (d) Information about consent from data providers/curators. [Not Applicable]
   (e) Discussion of sensible content if applicable, e.g., personally identifiable information or offensive content. [Not Applicable]

5. If you used crowdsourcing or conducted research with human subjects, check if you include:
   (a) The full text of instructions given to participants and screenshots. [Not Applicable]
   (b) Descriptions of potential participant risks, with links to Institutional Review Board (IRB) approvals if applicable. [Not Applicable]
   (c) The estimated hourly wage paid to participants and the total amount spent on participant compensation. [Not Applicable]