On the Construction and Implications of Low-Loss Valleys in LoRA-based Bayesian Inference

Daniel Dold[1], **Emanuel Sommer**[2,3], **Julius Kobialka**[2,3], **Oliver Dürr**[1,4], **David Rügamer**[2,3]
[1] HTWG Konstanz [2] LMU Munich [3] Munich Center for Machine Learning (MCML)
[4] Thurgau Institute for Digital Transformation (TIDIT)
{ddold, oliver.duerr}@htwg-konstanz.de
{emanuel.sommer, julius.kobialka, david.ruegamer}@stat.uni-muenchen.de

Abstract

While parameter-efficient fine-tuning methods like low-rank adaptation (LoRA) are standard for large language models, principled estimation of epistemic uncertainty remains challenging. Recent results in the LoRA regime suggest that discrete multi-mode approaches such as deep ensembles offer little benefit over single-mode methods. This contradicts broader observations in deep learning, where ensembling independent optima typically improves generalization, and linking these modes through continuous low-loss valleys further enhances Bayesian model averaging (BMA). Whether such structure exists in the LoRA space and whether it yields functional diversity missed by local or discrete methods has not been studied. We introduce *LoRA-Curve*, a segmented Bézier curve parameterization in the LoRA space, with two variants: a free configuration that jointly optimizes all control points, and an anchored configuration that connects independently fine-tuned LoRA optima. We establish pathwise continuity and regularity properties of the loss along the curve and empirically show, across multiple-choice reasoning and classification benchmarks with Qwen2.5 7B, that linear interpolation encounters loss barriers, while our anchored multi-segment curves connect independent optima through continuous low-loss valleys. Combined with flat-minima perturbations and a Jensen–Shannon divergence regularizer, *LoRA-Curve* yields measurably higher mutual information of the predictive distribution, indicating non-collapsed functional diversity along the curve, without sacrificing performance, and links continuous parameter-space traversal to functional diversity.

1 Introduction

Large language models (LLMs) have transformed language-based applications, yet they provide no principled estimate of epistemic uncertainty. This becomes especially problematic during fine-tuning, where point estimates lead to overconfident predictions and suboptimal generalization [see, e.g., 17]. To address this, probabilistic methods such as variational inference (VI) have been proposed; in the setting of low-rank adaptation (LoRA) [11], capturing a single, well-behaved mode has proven highly effective [22, 28]. The reported results in Wang et al. [28], Samplawski et al. [22] surprisingly show superior performance compared to deep ensembles [15], which contradicts broader findings where functional diversity stems from multiple distinct optima rather than a single basin. Existing Bayesian LoRA methods represent two extremes: either a single local mode (VI) or a discrete collection of isolated modes (DE). These optima can be connected by non-linear, low-loss paths [9, 8], forming a natural subspace for Bayesian model averaging (BMA) [7, 13].

Whether a continuous low-loss structure links these LoRA optima, and whether marginalizing over it provides functional diversity beyond these extremes, has not been studied in the context of LoRA.

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

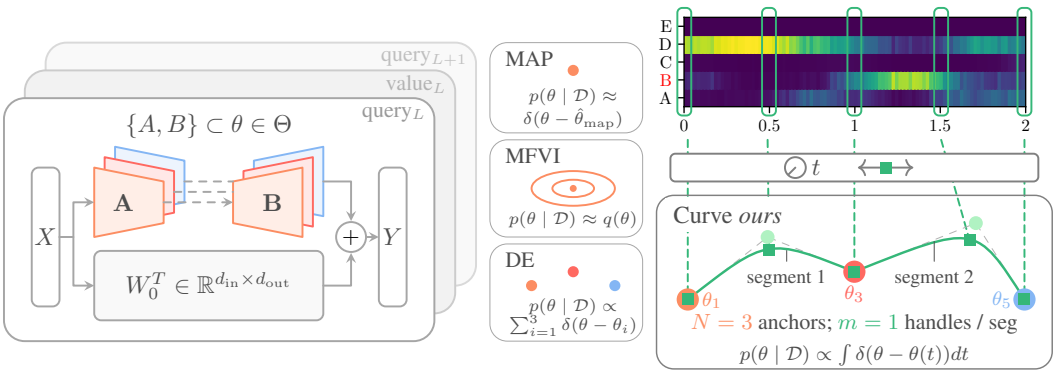


Figure 1: Overview of our LoRA-Curve approach. **Left:** Architecture for linear multiplication in an attention layer using low-rank adapters A and B, alongside frozen base weights W_0^T for different layers L. **Middle:** Comparison of Bayesian approximation strategies. Maximum A Posteriori (MAP) and mean-field variational inference (MFVI) capture only a single mode, whereas deep ensembles (DE) capture multiple distinct modes (colored orange, red, and blue). **Right:** Our proposed continuous parameterization. The framework connects $N = 3$ anchor points (e.g., DE modes) using a two-segment curve shaped by intermediate handles (green, $m = 1$ per segment), where anchors can either be fixed or jointly optimized with handles (bottom right). Mapping the curve parameter t reveals smooth, continuous transitions in predictive class probabilities between the discrete modes (top right). The green squares visualize the correspondence between the curve position and the resulting probabilities.

We therefore construct low-loss valleys in the LoRA weight space via Bézier curves initially used for mode connectivity [9], which simultaneously enable principled BMA.

This allows us to address the following research questions: 1) *Is it possible to construct meaningful low-loss valleys in the LoRA weight space?* 2) *Does continuous mode connectivity capture functional diversity missed by both local variational methods and discrete ensembles?* And 3) *What do these findings imply for probabilistic LoRA approaches?*

Contributions We thereby make the following contributions:

- We introduce *LoRA-Curve*, a framework for constructing continuous, segmented paths that explicitly enable smooth, functionally diverse traversals through the LoRA space.
- We demonstrate that continuous mode connectivity captures cross-modal functional diversity that discrete ensembles only partially exploit , while attaining competitive generalization performance.
- We characterize the resulting loss valleys and relate these findings to complementary approaches to understand the key properties of the LoRA posterior space.[1]

2 Background and related work

Fine-tuning LLMs is highly resource-intensive, prompting the widespread adoption of parameter-efficient fine-tuning methods like LoRA [11]. Given a layer with weights $W \in \mathbb{R}^{d_{\text{out}} \times d_{\text{in}}}$, LoRA [11] can be used to adapt a pre-trained weight $W_0 \in \mathbb{R}^{d_{\text{out}} \times d_{\text{in}}}$ by defining fine-tuning parameters $A \in \mathbb{R}^{r \times d_{\text{in}}}$ and $B \in \mathbb{R}^{d_{\text{out}} \times r}$ with dimension $r \ll \min(d_{\text{out}}, d_{\text{in}})$ and learning the update

$$W = W_0 + \Delta W, \qquad \Delta W = BA. \tag{1}$$

During the fine-tuning phase, only A and B matrices are optimized and W_0 is kept fixed. While this approach significantly reduces the number of trainable parameters, standard LoRA fine-tuning can still converge to sharp minima that generalize poorly [16]. To explicitly encourage the optimization process to settle in wide, generalizable regions of the loss landscape, Li et al. [16] suggested a perturbation strategy called Flat-LoRA, which extends Equation (1) by injecting noise ϵ during

[1] Experiments: https://github.com/doldd/Time_Transformer; lightweight standalone library: https://github.com/doldd/bayes_sub_inf.

training. The modified update rule then becomes $W = W_0 + \Delta W + \epsilon$ where $\epsilon \in \mathbb{R}^{d_{\text{out}} \times d_{\text{in}}}$ with entries $\epsilon_{i,j} \sim \mathcal{N}(0, \frac{\rho}{d_{\text{in}}} \|W'_{i,:}\|_2^2)$, $W'_{i,:}$ the ith row of the $W' = W_0 + \Delta W$, and hyperparameter $\rho > 0$. In the following, we will denote all (flattened) LoRA parameters, potentially gathered across multiple layers, with $\theta \in \Theta \subseteq \mathbb{R}^D$ with D indicating the number of trainable LoRA parameters.

Bayesian parameter-efficient fine-tuning Classic approaches for capturing predictive uncertainty in deep learning rely on variational inference (VI) [see, e.g., 2, 19], the Laplace approximation [e.g., 5], and deep ensembles (DEs) [15]. To scale these techniques to modern LLMs, recent work has integrated uncertainty estimation in the LoRA parameterization. Prominent variational examples include *Bayesian Low-Rank Adaptation by Backpropagation* (BLoB) [28], which learns a variational distribution over the A matrix via backpropagation, and *Scalable Bayesian Low-Rank Adaptation* (ScalaBL) [22], which utilizes further weight factorization to drastically reduce the stochastic parameter count for improved scalability. Other approaches include discrete LoRA ensembles to approximate the posterior [1, 27, 29], which we denote as DE in our experiments. Beyond variational and ensemble-based approximations, sampling-based inference has become increasingly practical for Bayesian neural networks, including approaches that combine independently initialized solutions with efficient posterior exploration [23–26].

Loss landscapes and mode connectivity While the previously mentioned approaches successfully capture epistemic uncertainty at a fraction of the full-parameter cost, their geometric assumptions are in stark contrast (single- vs. multi-mode) and conflict with each other regarding the geometry of the posterior or loss landscape. In general neural network applications, it has been shown that independent neural network optima are rarely isolated, but rather often form connected and continuous paths of low loss [8, 9]. This insight motivated cheap trajectory-based ensembles such as fast geometric ensembling (FGE) [9] and stochastic weight averaging (SWA) [12], which, however, do not explicitly optimize a connecting path. Recent work has further investigated the relationship between mode connectivity, posterior exploration, and Bayesian deep ensembles [24]. In particular, subspace-based posterior inference has been studied for semi-structured neural networks [6], while [7] investigate the geometry of low-loss paths and the role of prior specification for subspace inference. The analysis of low-loss valleys could therefore provide insights into the geometrical structure of LoRA posterior spaces, but so far, subspace methods have not been (successfully) applied to LoRA.

3 Constructing a curve-subspace for LoRA-based inference

We demonstrate how to efficiently construct continuous, low-loss valleys within the LoRA weight space. To this end, we introduce *LoRA-Curve*, a segmented Bézier parameterization for the LoRA space. Depending on the initialization and optimization of the anchor control points of the curve, we will distinguish between two configurations: **Free LoRA-Curve**, where all control points are jointly optimized, and **Anchored LoRA-Curve**, where segment boundaries (anchor points) are frozen to independently pretrained adapters (see Figure 1).

3.1 Free LoRA-Curve

Building on the flexibility of different parameterizations discussed in Garipov et al. [9], we construct the LoRA-Curve using a Bézier curve with $m+2$ control points. Specifically for the LoRA fine-tuning setting, we parameterize a continuous path $t \mapsto \zeta_\vartheta(t) \in \Theta$ for $t \in [0, 1]$ using

$$\zeta_\vartheta(t) = \textstyle\sum_{i=0}^{m+1} b_{i,(m+1)}(t)\theta_i,$$

where $\vartheta := \{\theta_0, \ldots, \theta_{(m+1)}\}$ represents the set of all control points $\theta_i \in \Theta$, and $b_{i,(m+1)}(t) = \binom{(m+1)}{i}(1-t)^{(m+1)-i}t^i$ are the Bernstein basis polynomials for a curve of degree $(m+1)$. Because each θ has a one-to-one mapping to the low-rank matrices A and B within a given layer, $\zeta_\vartheta(t)$ directly determines the continuous matrices $A_\vartheta(t)$ and $B_\vartheta(t)$. Consequently, the updated weight matrix at any point along the curve is $W_\vartheta(t) = W_0 + B_\vartheta(t)A_\vartheta(t)$. This curve defines an adaptation trajectory in the LoRA space: only A and B vary along t, while the pretrained weights W_0 remain fixed. Since all control points are jointly trained from scratch, we refer to this as the *Free LoRA-Curve* (FLC).

Optimization We train the curve by optimizing all control points ϑ using the expected loss along the curve

$$\mathcal{L}_{\text{curve}}(\vartheta) := \mathbb{E}_{t\sim U(0,1)}[\mathcal{L}_{\text{CE}}(\mathcal{D}; W_\vartheta(t))], \tag{2}$$

where $\mathcal{L}_{\text{CE}}$ is the negative categorical log-likelihood averaged over the training data $\mathcal{D}$. Unlike the standard continuous curve formulation of Garipov et al. [9], which optimizes a path between fixed, pre-trained endpoints, we jointly optimize the entire curve across all control points ϑ. To maintain standard LoRA behavior at initialization, we configure each control point θ_i by drawing the respective LoRA matrix A_i randomly and setting B_i to zero. This ensures $B_iA_i = 0$ for all i, so the initial weights evaluate exactly to W_0 everywhere along the curve.

During training, we approximate the expectation in Equation (2) by sampling a new $t \sim U(0, 1)$ for every minibatch. Compared to an $m + 2$-ensemble approach, our approach maintains an identical memory footprint but is less expensive to train, as it can update all curve members simultaneously and only requires a linear combination of control points to compute $\zeta_\vartheta(t)$, followed by a single forward pass per t-sample (the per-step FLOPs exceed those of a single LoRA update by less than 0.003%; cf. Table 4). Finally, to ensure the optimized curve settles in a flat low-loss valley, we incorporate the method of Li et al. [16] by injecting perturbation noise ϵ into W with $W = W_\vartheta(t) + \epsilon$.

Inference During inference for a new data point (x^*, y^*), we approximate the posterior predictive distribution via a temperature-scaled grid approximation along the optimized curve:

$$p(y^* \mid x^*, \mathcal{D}) \approx \sum_{j=1}^{M} \varpi_j(T)\, p\left(y^* \mid x^*, W_\vartheta(t_j), \mathcal{D}\right), \quad \varpi_j(T) = \frac{p(\mathcal{D} \mid t_j)^{1/T}}{\sum_{k=1}^{M} p(\mathcal{D} \mid t_k)^{1/T}},$$

where $\{t_j\}_{j=1}^M$ are M equispaced grid points. We empirically find that $T \to \infty$ yields the most robust performance across datasets. This corresponds to uniform averaging $\varpi_j = 1/M$. Izmailov et al. [13] already suggest downweighting the likelihood via tempering ($T > 1$) in subspace inference, since the full-data likelihood over a K-dimensional subspace of the D-dimensional parameter space concentrates the posterior far more strongly than in the full space. In our case this dimension mismatch is even larger ($K = 1$, $D \approx 7 \times 10^9$): small differences in the average log-likelihood along the curve are amplified exponentially by the number of data points, and the weights collapse onto few grid points. We therefore use the hyperparameter-free limit $T \to \infty$, which also outperforms validation-based stacking weights [31]. We provide further details in Appendix B.

3.2 Segmented curves

To navigate highly non-convex loss landscapes, a single polynomial segment is often insufficiently flexible (cf. Figure 5a). We therefore extend the LoRA-Curve framework to a composite Bézier curve comprising multiple individual segments. To construct this composite path, we define two distinct types of control points: *anchors*, which act as segment boundaries and lie directly on the curve, and *handles*, which are intermediate points that shape the curve but generally do not lie on it (cf. Figure 1).

Let N denote the number of anchors and m the number of handles per segment. For a curve with N anchor points connected by $N_{\text{seg}} = N - 1$ segments, each with m handles, the global parameter is $t \in [0, N_{\text{seg}}]$ and the total number of control points is $N_{\text{cp}} = N_{\text{seg}}(m + 1) + 1$. The active segment index is determined via $k = \min(\lfloor t \rfloor, N - 2) \in \{0, \ldots, N - 2\}$ with local coordinate $\tau = t - k \in [0, 1]$. We then evaluate the curve locally using this subset of control points $\zeta_\vartheta(\tau) = \sum_{i=0}^{m+1} b_{i,m+1}(\tau)\, \theta_{k(m+1)+i}$. This piecewise construction ensures that adjacent segments share exactly one boundary control point (an anchor point). This construction guarantees C^0 continuity while permitting sharp corners, allowing the path to flexibly bend around steep loss barriers (cf. the bend at θ_3 in Figure 1). By contrast, a single Bézier curve of degree $(N_{\text{cp}} - 1)$ is globally coupled: every control point influences the entire curve via its Bernstein basis function, making localized bends difficult and the path overly rigid when navigating highly non-convex valleys.

3.3 Anchored LoRA-Curve

While FLC optimizes all control points (the N anchor points and all $N_{\text{seg}}m$ handles) from scratch, we can explicitly guide the curve to connect known high-performing optima (e.g., those obtained

from a Deep Ensemble) by freezing the anchors to the weights of independently fine-tuned LoRA models. In this scenario, the remaining handles (initialized exactly as in FLC to explicitly increase diversity) are updated via gradient descent according to Equation (2). We call this approach *Anchored LoRA-curve (ALC)* and its configuration as ALC(N, m), where N is the number of frozen anchors and m the number of handles per segment. Analogously, FLC(N, m) denotes the free variant where the anchors and handles are jointly optimized.

This unified notation mathematically subsumes several standard methods as special cases: MAP corresponds to FLC$(1, 0)$ (a single point estimate with no curve); DE(N) corresponds to N independent anchors with $m = 0$; and ALC$(N, 0)$ reduces to standard piecewise linear mode connectivity between N optima, which we denote as Lin(N). ALC$(2, 1)$ recovers the curve of Garipov et al. [9], which they use to analyze the loss landscape; their FGE, in contrast, collects snapshots along a cyclical learning-rate trajectory without optimizing a connecting path.

3.4 Diversity regularization

To encourage functional diversity along the path and prevent the curve from collapsing into a single point-estimate, we employ a margin-based Jensen-Shannon Divergence (JSD) regularizer. Let $f(x; W_\vartheta(t))$ denote the model logits for input x at curve parameter $t \in [0, N_{\text{seg}}]$, and let $p_t := \text{softmax}(f(x; W_\vartheta(t)))$ denote the resulting predictive distribution. During each training step, we sample a point $t_1 \sim U(0, N_{\text{seg}})$ and compute an offset $t_2 = (t_1 + N_{\text{seg}}/2) \pmod{N_{\text{seg}}}$. This ensures that t_1 and t_2 are well separated along the curve's parameterization. We then compute the predictive distributions p_{t_1} and p_{t_2} and their mixture $\overline{p} = \frac{1}{2}(p_{t_1} + p_{t_2})$. The JSD between the two predictive distributions is $D_{\text{JSD}}(p_{t_1} \parallel p_{t_2}) = \frac{1}{2}(D_{\text{KL}}(p_{t_1} \parallel \overline{p}) + D_{\text{KL}}(p_{t_2} \parallel \overline{p}))$. To prevent unbounded JSD regularization from degrading primary task performance, we clip the penalty to τ_{JSD}. The total loss optimized at each step is then the symmetric cross-entropy loss augmented by this diversity penalty:

$$\mathcal{L} = \frac{1}{2}\Big(\mathcal{L}_{\text{CE}}(t_1) + \mathcal{L}_{\text{CE}}(t_2)\Big) + \lambda_{\text{JSD}}\mathcal{L}_{\text{JSD}},$$

where λ_{JSD} controls the regularization strength. By explicitly penalizing the similarity of predictions at distant points on the curve, the JSD regularizer forces the trajectory to explore diverse functional regions of the loss valley.

4 Properties of Bayesian inference using LoRA-Curve

While connecting well-performing solutions might improve the quality of performance and uncertainty quantification, we see the primary merit of the curve in its properties, which allow us to connect the parameter to the output space and interpret the model's posterior geometry. The following propositions are direct consequences of the construction; we state them as supporting properties rather than as a central theoretical contribution.

4.1 Continuity

In contrast to DEs or samples from a variational distribution that only provide access to a finite set of isolated predictive distributions $\{p(y \mid x, \theta_i)\}_{i=1}^N$, the construction of a curved subspace allows users to understand how predicted probabilities update when making changes in the LoRA weight space. A straightforward but important result is the following.

Proposition 1. *The output distribution of the LoRA-Curve $p(y|x, W_\vartheta(t), \mathcal{D})$ is continuous in t*

This result follows as the composition

$$t \;\mapsto\; \zeta_\vartheta(t) \;\mapsto\; W_0 + B_\vartheta(t)\, A_\vartheta(t) \;\mapsto\; f\big(x; W_\vartheta(t)\big) \;\mapsto\; p(y|x, W_\vartheta(t), \mathcal{D})$$

is continuous in each step: $\zeta_\vartheta(t) = \sum_{i=0}^{m+1} b_{i,m+1}(t)\, \theta_i$ is polynomial in t (hence C^∞); the recovery of $A_\vartheta(t)$ and $B_\vartheta(t)$ from $\zeta_\vartheta(t)$ is simply a projection onto fixed index subsets of $\mathbb{R}^D$, and is therefore continuous; the product $B_\vartheta(t)\, A_\vartheta(t)$ is bilinear and continuous in both factors; the forward pass $f(\,\cdot\,; W)$ is continuous in W for activations such as SiLU used in LLMs; and the softmax is continuous. Furthermore, for segmented curves, C^0 continuity at anchor points holds by construction, since adjacent segments share a boundary control point (cf. Section 3.2).

As depicted in Figure 3, this property provides direct access to the roles of weights in the LoRA space by relating the change in the low-loss valley to the change in output probabilities. The subspace thus not only provides a way to perform probabilistic inference in the LoRA space, but can also be used to interpret the importance and role of different weights.

4.2 Posterior geometry along the curve

Since posterior geometry in overparameterized neural networks is affected by non-identifiability and the interaction between priors and parameter redundancy [14, 20], variation in parameter space need not imply corresponding functional diversity. While the successful application of ALC and FLC as demonstrated in the next section suggests the existence of a continuous low-loss valley, continuity alone does not quantify how predictions and losses change along the curve. We therefore study the empirical risk restricted to a fixed LoRA-Curve, $\ell_\vartheta(t) := \mathcal{L}_{\mathrm{CE}}(\mathcal{D}; W_\vartheta(t)), t \in [0, N_{\mathrm{seg}}]$. Rather than requiring the restricted loss to be piecewise polynomial, which is not generally true for modern architectures involving softmax, SiLU, GELU, or normalization layers, we only require the regularity of the composed neural network map.

Proposition 2. *Assume that the LoRA-Curve $t \mapsto W_\vartheta(t)$ is piecewise continuously differentiable on $[0, N_{seg}]$ and that, for every datapoint $(x, y) \in \mathcal{D}$, the map $W \mapsto \mathcal{L}_{\mathrm{CE}}(y, f(x; W))$ is continuously differentiable on an open neighborhood of the curve image $\{W_\vartheta(t) : t \in [0, N_{seg}]\}$. Then the restricted curve loss ℓ_ϑ is continuous and piecewise continuously differentiable. On every differentiable segment, $\ell'_\vartheta(t) = \left\langle \nabla_W \mathcal{L}_{\mathrm{CE}}(\mathcal{D}; W_\vartheta(t)), \frac{d}{dt} W_\vartheta(t) \right\rangle$. Moreover, since the curve image is compact, ℓ_ϑ is Lipschitz on each segment and absolutely continuous on $[0, N_{seg}]$. In particular, for all s, t in the same segment,*

$$|\ell_\vartheta(t) - \ell_\vartheta(s)| \leq \int_s^t \|\nabla_W \mathcal{L}_{\mathrm{CE}}(\mathcal{D}; W_\vartheta(u))\| \, \|\frac{d}{du} W_\vartheta(u)\| du.$$

This result follows directly from the chain rule and compactness. The LoRA-Curve is piecewise continuously differentiable because it is constructed from polynomial curve segments in the low-dimensional LoRA parameter space, followed by the affine-bilinear map $(A, B) \mapsto W_0 + BA$. Modern LLM components such as affine maps, matrix products, and SiLU or softmax activations are continuous and smooth on their natural domains. Therefore, the loss restricted to the LoRA-Curve inherits this regularity.

The proposition provides a conservative but useful sense in which the LoRA-Curve exposes local posterior geometry. The loss cannot jump along the curve, and its change between two curve locations is controlled by two quantities: the norm of the full loss gradient and the speed with which the curve moves through weight space. Thus, sharp changes in prediction or loss along the curve must be explained either by large local sensitivity of the model, measured by $\|\nabla_W \mathcal{L}_{\mathrm{CE}}(\mathcal{D}; W_\vartheta(t))\|$, or by rapid movement of the LoRA-Curve in parameter space, measured by the curve's speed $\|\frac{d}{dt} W_\vartheta(t)\|$.

From an inference perspective, this means that predictive distributions sampled along the curve are not isolated points, as in deep ensembles, but form a continuous, pathwise-regular trajectory. The curve, therefore, provides an interpretable one-dimensional probe of the local posterior geometry: it connects high-performing parameter settings while making changes in predictive probabilities attributable to movement in the LoRA space.

5 Experimental study

To answer our core research questions, our evaluation proceeds in two stages. First, we analyze the geometric properties of the LoRA loss landscape to understand the structural implications of the low-loss paths we have discovered. Second, we inspect how the construction and regularization of these curves directly influence downstream generalization performance. To this end, we evaluate our Anchored (ALC) and Free (FLC) LoRA-Curve frameworks against single-mode baselines (MAP), the VI approaches BLoB [28] and ScalaBL [22], discrete multi-mode LoRA ensembles (DE), and the trajectory-based ensembles FGE and SWA. We fine-tune Qwen 2.5 7B [30] via LoRA across standard reasoning and classification benchmarks (ARC-Challenge, ARC-Easy [4], OBQA [18], boolq [3], Winogrande (Small/Medium)[21]). We select this scale to align with prior work [22] and avoid capacity saturation. Larger models already handle these benchmarks confidently, leaving

little epistemic uncertainty for LoRA fine-tuning to capture. Our study is thus restricted to a single backbone and to multiple-choice classification tasks. We use LoRA rank $r = 8$ and ablate $r = 40$ in Section D.1. Most hyperparameters follow Wang et al. [28], Samplawski et al. [22]; only the Flat-LoRA and JSD parameters are tuned on Winogrande-S.

Details on hyperparameter choices and our training routine can be found in Section F, a performance comparison in Section C, and further experimental results in Section D.

5.1 Properties of the LoRA loss-landscape

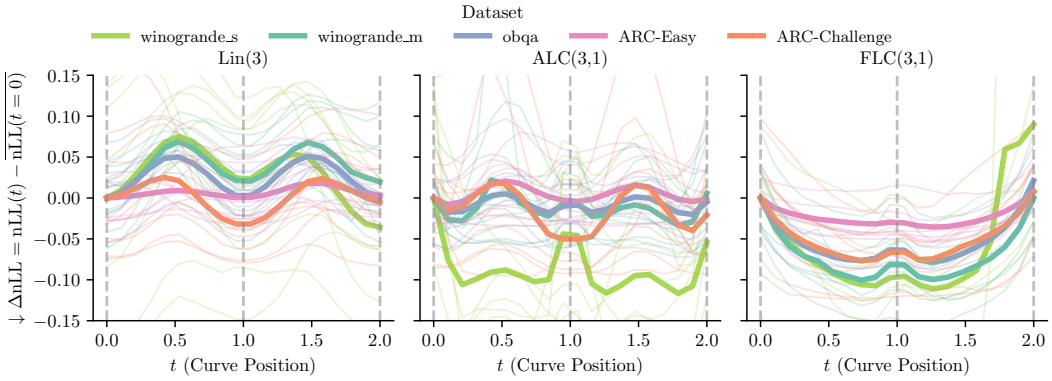


Figure 2: Mode connectivity comparison of linear vs. segmented curves visualized by changes in the negative log-likelihood (Δ nLL) on training data relative to the starting point along the interpolation path. Vertical dashed lines denote the locations of the anchors. Thick solid lines represent the mean across eight repetitions for each dataset (colors), and faded lines show individual runs. Although linear interpolation encounters distinct loss barriers, the ALC and especially the FLC parameterizations effectively circumvent these barriers to navigate continuous, low-loss valleys.

To better understand the geometry of the LoRA space, we first examine whether independent optima can be connected via low-loss paths. Figure 2 compares linear interpolation against our segmented curve approaches. Linear interpolation (Linear(3)) encounters distinct loss barriers, manifesting as twin peaks midway between each segment at $t = 0.5$ and $t = 1.5$. Although the anchored curve (ALC(3,1)) reduces these barriers, strictly connecting fixed, pretrained anchors remains challenging because they may lie in suboptimal positions for continuous traversal. By contrast, the free LoRA-Curve (FLC(3,1)) easily navigates these valleys on the training data. By jointly optimizing all control points, FLC dynamically shifts the anchors to more favorable locations, thereby constructing smoother, low-loss connections. The overall improved flatness along the interior of the FLC path demonstrates that the LoRA loss landscape between different optima is inherently non-convex, necessitating flexible parameterizations beyond simple linear interpolation.

Figure 3 illustrates the continuity property established in Proposition 1. Unlike DE(5) and BLoB, which exhibit discrete jumps at every mode or between samples from their variational distributions, our curve parameterization guarantees smooth transitions in logit space. This provides a principled mechanism for linking continuous parameter-space traversal to functional diversity, while intermediate models still remain highly performant. This continuity is particularly striking when comparing DE(5) to ALC(5,1). Because ALC(5,1) connects these exact modes (e.g., the curve at $t = 0$ corresponds to the first ensemble member $i = 1$), it bridges the predictive gaps: while isolated DE members offer no insight into the functional space between them, ALC explicitly reveals the smooth transitions bridging these optima. The same smooth transitions persist on constructed OOD questions with several or no correct answers, where the curve makes visible how the predictive distribution moves between optima, although all methods, including ours, fail on some of these edge cases (Section D.2).

Beyond just smoothness, our segmented Bézier parameterization leverages intermediate control points to actively inject functional diversity, in stark contrast to the rigid interpolation of linear mode connectivity. This is visually evident again in Figure 3, where the ALC(5,1) class probabilities shift dynamically at each new intermediate control point (i.e., at every 0.5 increment of t). Consequently,

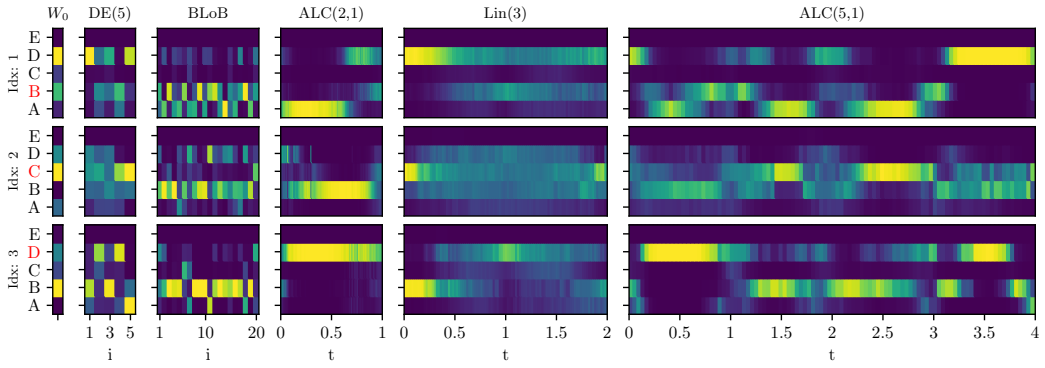


Figure 3: Class probability evolution across discrete and continuous methods, using a linear color scale where dark blue represents a probability of 0 and yellow represents 1 for the five labels A–E. Different rows correspond to three test examples from the ARC-Challenge dataset. Red y-axis labels indicate the ground-truth class. Facet columns represent different approaches with the base Qwen 2.5 model without LoRA pretraining, denoted as W_0. The x-axis represents either the discrete model index i or the continuous curve parameter t.

curves parameterized with a higher density of control points inherently generate richer functional diversity. We quantitatively validate this in Figure 5d.

5.2 Implications of low-loss valleys design on the performance

Figure 4 presents a direct performance comparison between our curve approaches and discrete DEs. Because ALC uses the exact same anchors as DE, it provides a controlled framework that isolates both the effect of continuous versus discrete averaging (comparing DE(N) to Lin(N)) and the necessity of non-linear routing (comparing ALC(N, m) to Lin(N)). The linear formulation (ALC with $m = 0$) proves too restrictive, failing to capture the curvature of the underlying low-loss manifold. By contrast, the higher-order ALC$(N, 2)$ successfully matches or outperforms DE across datasets.

FLC, however, exhibits less consistent gains, struggling particularly on the `winogrande_m` dataset. This performance gap is intriguing because FLC and ALC possess identical representational capacity; in principle, FLC can represent any curve that ALC can. The discrepancy, therefore, stems strictly from the optimization strategy. While jointly optimizing all control points makes FLC significantly faster to train overall (cf. Table 6), it induces fundamentally different optimization dynamics compared to the sequential ALC process. Furthermore, this exposes a prominent generalization gap. Although Figure 2 demonstrates that FLC discovers smooth, low-loss valleys in the training data, this does not consistently translate into superior test performance, indicating a misalignment between the training and test landscapes (cf. Figure 8). The FLC valleys are nevertheless not an artifact of a collapsed curve, as its MI and log-likelihood grow with N similarly to ALC (Table 1). The single-trajectory baselines remain weaker: SWA and FGE(17) stay below DE(9) and ALC(5,2) in log-likelihood on all datasets (Table 1). Figure 15 shows that one of the curve configurations always lies on the Pareto front of training cost (measured in optimizer steps) and test log-likelihood for every dataset, both for the full step budget and for the steps until the best validation checkpoint. Section G.4 furthermore reports the inference cost. Despite these optimization challenges, increasing the number of anchors N generally improves performance for all methods (DE, ALC, and FLC). This suggests that multiple modes, as captured by longer paths, result in better performance regardless of the specific training regimen.

Figure 5a examines the necessity of segmentation by comparing single-segment curves against multi-segment curves, grouped by an equivalent total number of control points N_{cp}. Multi-segment configurations consistently match or outperform their single-segment counterparts. Furthermore, this performance advantage becomes even more pronounced as the total number of control points increases (cf. Figures 5a and 10).

Figure 5b investigates whether the discovered low-loss paths also benefit from wider valleys. By combining our curve framework with Flat LoRA weight perturbations ($\rho > 0$), we demonstrate

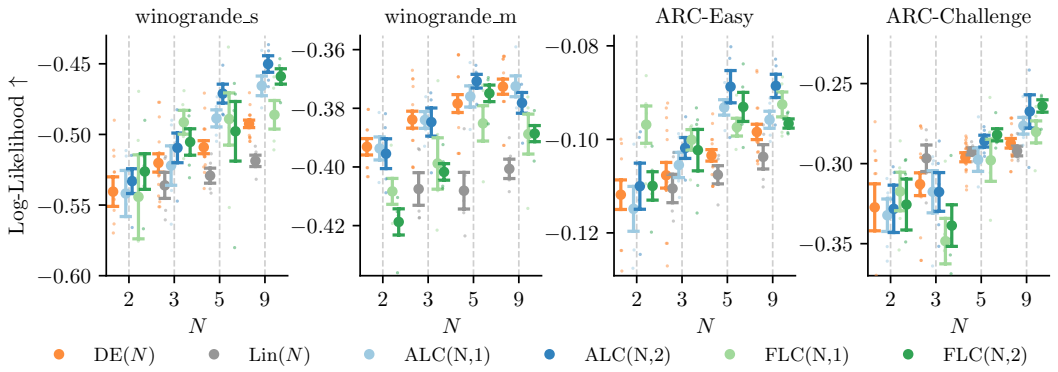


Figure 4: Comparison of our LoRA-Curve framework against DEs using test log-likelihood (↑) across four datasets (panels; `obqa` and `boolq` in Figure 11) evaluated for varying numbers of anchors (N, x-axis). Colors distinguish discrete DE (orange) from our continuous curve parameterizations. Solid points with error bars denote the mean and standard error across 5 random seeds, shown as faded dots.

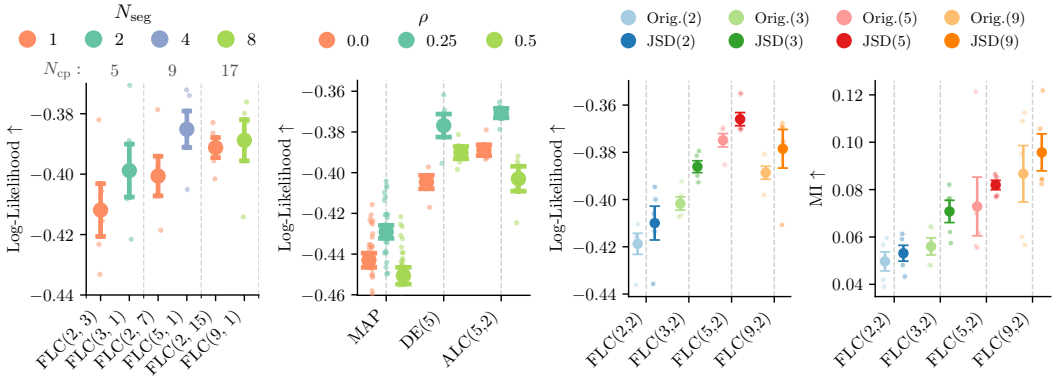
that optimization objectives targeting flatter regions successfully force continuous paths to lie in broader regions of the loss landscape. Consequently, the generalization benefits of locating these flatter optima—as observed in the MAP and DE baselines—transfer directly to our continuous curve framework. However, regardless of the underlying method, excessive noise injection (e.g., $\rho = 0.5$) ultimately masks the training signal, indicating that the optimal perturbation strength requires careful tuning.

Finally, Figures 5c and 5d evaluate the impact of JSD regularization on performance and functional diversity. As detailed in Section 3.4, this regularizer explicitly forces the continuous path to explore functionally distinct regions. Figure 5c shows that JSD-regularized models (darker shades) maintain, or even improve, log-likelihoods compared to their unregularized baselines (lighter shades). Furthermore, Figure 5d demonstrates that mutual information (MI; cf. Section A) increases through two compounding mechanisms: the addition of more anchor points N and the explicit application of JSD regularization. In preliminary experiments, a repulsive regularizer that pushes control points apart in weight space neither increased functional diversity nor improved performance (Section E.1).

6 Conclusion

In this work, we introduced *LoRA-Curve*, a segmented Bézier parameterization designed to investigate the continuous posterior geometry of LLMs fine-tuned via low-rank adaptation. Contrary to recent suggestions that the LoRA landscape lacks beneficial multi-mode structure, our findings demonstrate that the space between independent optima is a rich, navigable region of low loss. Our free (FLC) and anchored (ALC) configurations successfully construct meaningful, continuous low-loss valleys that bypass steep loss barriers. Furthermore, these two variants offer valuable practical trade-offs: FLC provides an efficient optimization approach by dynamically discovering valleys from scratch, whereas ALC enables the explicit, targeted connection of independently fine-tuned optima.

Navigating these non-convex valleys captures unique functional diversity. Unlike local variational methods that explore only a single basin, or discrete DEs that aggregate isolated point estimates, *LoRA-Curve* guarantees smooth predictive transitions and explicitly links continuous parameter-space traversal to measurably higher mutual information without sacrificing predictive performance. For probabilistic LoRA applications, our results establish that the continuous space between modes serves as a viable manifold for BMA. For a fixed number of control points, splitting the curve into segments matches or outperforms a single high-degree curve, and the functional diversity along these valleys grows with the number of anchors and with output-space (JSD) regularization. Combining this parameterization with flat-minima perturbations broadens the loss valley occupied by the curve, while JSD regularization further increases functional diversity along it.



(a) Curve vs. Segments (b) Flat Curve (c) JSD (LL) (d) JSD (MI)

Figure 5: Performance and diversity analysis on the `winogrande_m` dataset. **(a) Curve vs. segments:** Test log-likelihood (↑) comparison of single-segment curves (orange) versus multi-segment curves (blue and greens), grouped by an equivalent number of control points N_{cp}. **(b) Flat Curve:** The effect of Flat LoRA weight perturbations ($\rho > 0$, greens) compared to the unperturbed baseline ($\rho = 0$, orange) for different approaches. **(c) & (d) JSD Regularization:** The impact of JSD regularization on log-likelihood and MI (↑), respectively. Darker shades denote JSD-regularized models, and lighter shades represent the unregularized (Orig.) baselines across varying FLC complexities. Across all panels, solid points with error bars indicate the mean and standard error, and faded dots represent individual runs.

Limitations While the *LoRA-Curve* framework provides a principled approach to continuous mode connectivity, it presents certain structural and practical limitations. First, we restrict our analysis to one-dimensional, globally coupled trajectories. Extending this parameterization to higher-dimensional surfaces or assigning independent curves to individual LoRA adapters would reduce interpretability; in preliminary experiments, the latter also did not improve performance. Furthermore, BMA over continuous manifolds generally requires computing posterior weights, which is computationally expensive. Along our curves, likelihood-based weights ($T = 1$) concentrate on few grid points, so we use uniform weighting ($T \to \infty$), a tempering choice motivated by the extreme mismatch between the 1-D subspace and the full parameter dimension and validated empirically (see Section B), which also avoids computing the weights in our one-dimensional case. However, in higher-dimensional parameterizations, uniform weighting would likely prove insufficient, thereby reintroducing the prohibitive cost of evaluating the full posterior. While the free configuration (FLC) accelerates optimization, it currently exhibits a prominent generalization gap. At inference, both curve variants evaluate M grid points instead of N members, although M can be reduced with little loss (Section G.4). ALC additionally requires the anchor training of a DE plus a curve optimization and is thus most attractive when a DE is already available, whereas FLC offers a cheaper alternative in training. Higher MI indicates non-collapsed diversity along the curve but does not imply better calibration, where VI and DE baselines remain competitive. Finally, our experiments cover a single backbone and multiple-choice classification tasks only.

Acknowledgments and Disclosure of Funding

This research was supported by grants from NVIDIA and utilized NVIDIA products, including NVIDIA A100 and NVIDIA RTX 6000 GPUs. DR's research is funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) – 578966082. DD's and OD's research is funded by the Carl-Zeiss-Stiftung in the project "DeepCarbPlanner" (grant no. P2021-08-007).

References

[1] Oleksandr Balabanov and Hampus Linander. Uncertainty quantification in fine-tuned LLMs using LoRA ensembles. In *ICLR Workshop: Quantify Uncertainty and Hallucination in Foundation Models: The Next Frontier in Reliable AI*, 2025.

[2] Charles Blundell, Julien Cornebise, Koray Kavukcuoglu, and Daan Wierstra. Weight Uncertainty in Neural Network. In *Proceedings of the 32nd International Conference on Machine Learning*, pages 1613–1622. PMLR, June 2015.

[3] Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. Boolq: Exploring the surprising difficulty of natural yes/no questions. In *Proceedings of the 2019 conference of the north American chapter of the association for computational linguistics: Human language technologies, volume 1 (long and short papers)*, pages 2924–2936, 2019.

[4] Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv preprint arXiv:1803.05457*, 2018.

[5] Erik Daxberger, Agustinus Kristiadi, Alexander Immer, Runa Eschenhagen, Matthias Bauer, and Philipp Hennig. Laplace Redux - Effortless Bayesian Deep Learning. In *Advances in Neural Information Processing Systems*, volume 34, pages 20089–20103. Curran Associates, Inc., 2021.

[6] Daniel Dold, David Rügamer, Beate Sick, and Oliver Dürr. Bayesian semi-structured subspace inference. In *Proceedings of The 27th International Conference on Artificial Intelligence and Statistics*, volume 238 of *Proceedings of Machine Learning Research*, pages 1819–1827. PMLR, 2024.

[7] Daniel Dold, Julius Kobialka, Nicolai Palm, Emanuel Sommer, David Rügamer, and Oliver Dürr. Paths and Ambient Spaces in Neural Loss Landscapes. In *Proceedings of The 28th International Conference on Artificial Intelligence and Statistics*, pages 10–18. PMLR, April 2025.

[8] Felix Draxler, Kambis Veschgini, Manfred Salmhofer, and Fred Hamprecht. Essentially no barriers in neural network energy landscape. In *International conference on machine learning*, pages 1309–1318. PMLR, 2018.

[9] Timur Garipov, Pavel Izmailov, Dmitrii Podoprikhin, Dmitry P Vetrov, and Andrew G Wilson. Loss Surfaces, Mode Connectivity, and Fast Ensembling of DNNs. In *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018.

[10] Neil Houlsby, Ferenc Huszár, Zoubin Ghahramani, and Máté Lengyel. Bayesian active learning for classification and preference learning. *arXiv preprint arXiv:1112.5745*, 2011.

[11] Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-Rank Adaptation of Large Language Models. In *ICLR 2022*, April 2022.

[12] Pavel Izmailov, Dmitrii Podoprikhin, Timur Garipov, Dmitry Vetrov, and Andrew Gordon Wilson. Averaging weights leads to wider optima and better generalization. In *Proceedings of the 34th Conference on Uncertainty in Artificial Intelligence*, pages 876–885, 2018.

[13] Pavel Izmailov, Wesley J. Maddox, Polina Kirichenko, Timur Garipov, Dmitry Vetrov, and Andrew Gordon Wilson. Subspace Inference for Bayesian Deep Learning. In *Proceedings of The 35th Uncertainty in Artificial Intelligence Conference*, pages 1169–1179. PMLR, August 2020.

[14] Julius Kobialka, Emanuel Sommer, Chris Kolb, Juntae Kwon, Daniel Dold, and David Rügamer. On the interplay of priors and overparametrization in Bayesian neural network posteriors. In *Proceedings of The 29th International Conference on Artificial Intelligence and Statistics*, volume 300 of *Proceedings of Machine Learning Research*, pages 19–27. PMLR, 2026.

[15] Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and Scalable Predictive Uncertainty Estimation using Deep Ensembles. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

[16] Tao Li, Zhengbao He, Yujun Li, Yasheng Wang, Lifeng Shang, and Xiaolin Huang. Flat-LoRA: Low-Rank Adaptation over a Flat Loss Landscape. In *Proceedings of the 42nd International Conference on Machine Learning*, pages 34549–34563. PMLR, October 2025.

[17] Yawei Li, David Rügamer, Bernd Bischl, and Mina Rezaei. Calibrating LLMs with information-theoretic evidential deep learning. In *The Thirteenth International Conference on Learning Representations*, 2025.

[18] Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. Can a suit of armor conduct electricity? a new dataset for open book question answering. In *Proceedings of the 2018 conference on empirical methods in natural language processing*, pages 2381–2391, 2018.

[19] Rajesh Ranganath, Sean Gerrish, and David Blei. Black Box Variational Inference. In *Proceedings of the Seventeenth International Conference on Artificial Intelligence and Statistics*, pages 814–822. PMLR, April 2014.

[20] David Rügamer. On the epistemic uncertainty of overparametrized neural networks. In *Proceedings of the 43rd International Conference on Machine Learning*, volume 306 of *Proceedings of Machine Learning Research*, pages 106165–106185. PMLR, 2026.

[21] Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. Winogrande: An adversarial winograd schema challenge at scale. *Communications of the ACM*, 64(9):99–106, 2021.

[22] Colin Samplawski, Adam D Cobb, Manoj Acharya, Ramneet Kaur, and Susmit Jha. Scalable Bayesian Low-Rank Adaptation of Large Language Models via Stochastic Variational Subspace Inference. In *Conference on Uncertainty in Artificial Intelligence*, pages 3587–3604. PMLR, 2025.

[23] Emanuel Sommer and David Rügamer. Position: The time for sampling is now! Charting a new course for Bayesian deep learning. In *Proceedings of the 43rd International Conference on Machine Learning*, volume 306 of *Proceedings of Machine Learning Research*, pages 172098–172111. PMLR, 2026.

[24] Emanuel Sommer, Lisa Wimmer, Theodore Papamarkou, Ludwig Bothmann, Bernd Bischl, and David Rügamer. Connecting the dots: Is mode-connectedness the key to feasible sample-based inference in Bayesian neural networks? In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 45988–46018. PMLR, 2024.

[25] Emanuel Sommer, Jakob Robnik, Giorgi Nozadze, Uros Seljak, and David Rügamer. Microcanonical Langevin ensembles: Advancing the sampling of Bayesian neural networks. In *International Conference on Learning Representations*, 2025.

[26] Emanuel Sommer, Kangning Diao, Jakob Robnik, Uros Seljak, and David Rügamer. Can microcanonical Langevin dynamics leverage mini-batch gradient noise? In *Proceedings of the 43rd International Conference on Machine Learning*, volume 306 of *Proceedings of Machine Learning Research*, pages 114383–114412. PMLR, 2026.

[27] Xi Wang, Laurence Aitchison, and Maja Rudolph. LoRA ensembles for large language model fine-tuning. *arXiv preprint arXiv:2310.00035*, 2023.

[28] Yibin Wang, Haizhou Shi, Ligong Han, Dimitris Metaxas, and Hao Wang. BLoB: Bayesian Low-Rank Adaptation by Backpropagation for Large Language Models. *Advances in Neural Information Processing Systems*, 37:67758–67794, December 2024. doi: 10.52202/079017-2164.

[29] Adam X. Yang, Maxime Robeyns, Xi Wang, and Laurence Aitchison. Bayesian Low-rank Adaptation for Large Language Models. In *The Twelfth International Conference on Learning Representations*, 2024.

[30] An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, et al. Qwen2. 5 technical report. *arXiv preprint arXiv:2412.15115*, 2024.

[31] Yuling Yao, Aki Vehtari, Daniel Simpson, and Andrew Gelman. Using stacking to average Bayesian predictive distributions (with discussion). *Bayesian Analysis*, 13(3):917–1007, 2018. doi: 10.1214/17-BA1091.

A Mutual information (MI)

To quantify epistemic uncertainty as a measure of functional diversity within our LoRA-Curve framework, we compute the Mutual Information (MI) between the curve parameter t and the predictive output y, following the Bayesian Active Learning by Disagreement (BALD) objective [10]. Assuming a distribution $p(t)$ over the curve parameter $t \in [0, T_{\max}]$, the theoretical continuous MI for a single input $\mathbf{x}$ is defined as:

$$I[y, t \mid \mathbf{x}, \mathcal{D}] \equiv \mathrm{MI}(\mathbf{x}) = \mathrm{H}\left[\mathbb{E}_{p(t)}[p(y \mid \mathbf{x}, \theta_\vartheta(t))]\right] - \mathbb{E}_{p(t)}\left[\mathrm{H}[p(y \mid \mathbf{x}, \theta_\vartheta(t))]\right]$$

where $\theta_\vartheta(t)$ denotes the model parameters at curve position t.

In practice, we approximate this continuous objective via numerical integration:

$$\mathrm{MI}(\mathbf{x}) \approx \underbrace{\mathrm{H}\left[\sum_{j=1}^{M} w_j\, p(y \mid \mathbf{x}, \theta_\vartheta(t_j))\right]}_{\text{Total Uncertainty}} - \underbrace{\sum_{j=1}^{M} w_j\, \mathrm{H}[p(y \mid \mathbf{x}, \theta_\vartheta(t_j))]}_{\text{Expected Data Uncertainty}}$$

where $\{t_j\}_{j=1}^{M}$ represents $M = 2N_{\mathrm{cp}} - 1$ equispaced grid points along the curve (Section F) and w_j represent numerical quadrature weights (defaulting to uniform weights $w_j = 1/M$ for a uniform posterior $p(t|\mathcal{D})$).

In our implementation, we evaluate this objective by first calculating the total predictive uncertainty across the entire curve. This is achieved by computing the weighted average predictive distribution $\bar{p} = \sum_{j=1}^{M} w_j\, p_j$, where $p_j = \mathrm{softmax}(f(\mathbf{x}; \theta_\vartheta(t_j)))$, and evaluating its entropy $H[\bar{p}]$. We then capture the expected pointwise (aleatoric) uncertainty by computing the weighted average of the individual prediction entropies, $\sum_j w_j H[p_j]$. The epistemic uncertainty (the MI score) is subsequently defined as the difference between these two quantities: $H[\bar{p}] - \sum_j w_j H[p_j]$. Consequently, a high MI score indicates that predictions vary significantly along the curve—reflecting high functional disagreement between different t values—even if individual parameter configurations remain confident. This effectively isolates the functional diversity captured by the continuous trajectory.

Finally, the single scalar MI metric reported in our experiments represents the mean mutual information computed across all datapoints in the evaluation dataset $\mathcal{D}$:

$$\mathrm{MI}_{\text{reported}} = \frac{1}{|\mathcal{D}|} \sum_{\mathbf{x} \in \mathcal{D}} \mathrm{MI}(\mathbf{x})$$

B Temperature scaling and Bayesian model averaging

In our setting, Bayesian model averaging (BMA) requires evaluating the posterior probability at discrete grid points $\{t_j\}_{j=1}^{M}$ on the training dataset $\mathcal{D}$. We formulate this using a grid approximation over a uniform prior $t \sim U(0, N_{\text{seg}})$ with a temperature scaling parameter T:

$$p(y^* \mid x^*, \mathcal{D}) \approx \sum_{j=1}^{M} \varpi_j(T)\, p\left(y^* \mid x^*, W_\vartheta(t_j), \mathcal{D}\right), \quad \varpi_j(T) = \frac{p(\mathcal{D} \mid t_j)^{1/T}}{\sum_{k=1}^{M} p(\mathcal{D} \mid t_k)^{1/T}},$$

Optimizing the temperature on a validation set reveals that uniform averaging, corresponding to an infinite temperature ($T \to \infty$), yields the best empirical performance. This is justified mathematically by the limit:

$$\lim_{T \to \infty} \varpi_j(T) = \lim_{T \to \infty} \frac{p(\mathcal{D} \mid t_j)^{1/T} p(t_j)}{\sum_{k=1}^{M} p(\mathcal{D} \mid t_k)^{1/T} p(t_k)} = \frac{1}{M}.$$

Dimension mismatch and posterior concentration Izmailov et al. [13] perform Bayesian inference in a K-dimensional subspace of the D-dimensional parameter space and downweight the likelihood via temperature scaling ($T > 1$): the full-data likelihood is retained while the number of parameters is drastically reduced, so the subspace posterior is far more concentrated than the

corresponding posterior in the full space. Our curve is an extreme instance of this mismatch, a $K = 1$ subspace of a model with $D \approx 7 \times 10^9$ parameters. With $n = |\mathcal{D}|$ and the average log-likelihood $\bar{\ell}(t)$, the weights satisfy $\varpi_j(T) \propto \exp(n\,\bar{\ell}(t_j)/T)$. Even small variations of $\bar{\ell}(t)$ along the curve (cf. Figure 2) are therefore amplified exponentially. A gap of $\Delta\bar{\ell} = 0.01$ yields weight ratios of order $e^{n/100}$ at $T = 1$, and the average collapses to an argmax over the grid for large n, a known behavior of likelihood-proportional weighting when the true data-generating process is not among the candidate models [31]. Since the mismatch in our setting is even larger than in Izmailov et al. [13], we choose the hyperparameter-free limit $T \to \infty$. Computing the posterior without temperature scaling ($T = 1$) thus degrades performance back towards single-point estimates (cf. Figure 6).

Validation-based stacking As a data-splitting alternative, we fit weights w for the M grid points on the validation data via stacking [31], which maximizes the validation log-score of the mixture $\sum_j w_j\, p_{ij}$ over the probability simplex, with p_{ij} the predictive probability of the correct answer for validation example i at t_j. We quantify the spread of the weights by the effective sample size $\mathrm{ESS}(w) = \exp(-\sum_j w_j \log w_j) \in [1, M]$, i.e., the effective number of grid points receiving weight. Across all six datasets, ALC and FLC, and 5 seeds, we find $\mathrm{ESS}/M \approx 0.06$ for the $T = 1$ posterior (computed on training and validation data) and ≈ 0.17 for stacking, far from uniform averaging ($\mathrm{ESS}/M = 1$). On the test data, uniform averaging achieves a higher log-likelihood than both the $T = 1$ posterior and stacking in 11 of 12 dataset/configuration settings, with the remaining one within seed variability. Uniform averaging is moreover the convention of the discrete counterparts we compare against (DE, FGE). Consequently, we discard standard temperature scaling and approximate the Bayesian predictive distribution via the uniform averaging ($T \to \infty$) defined in the main text.

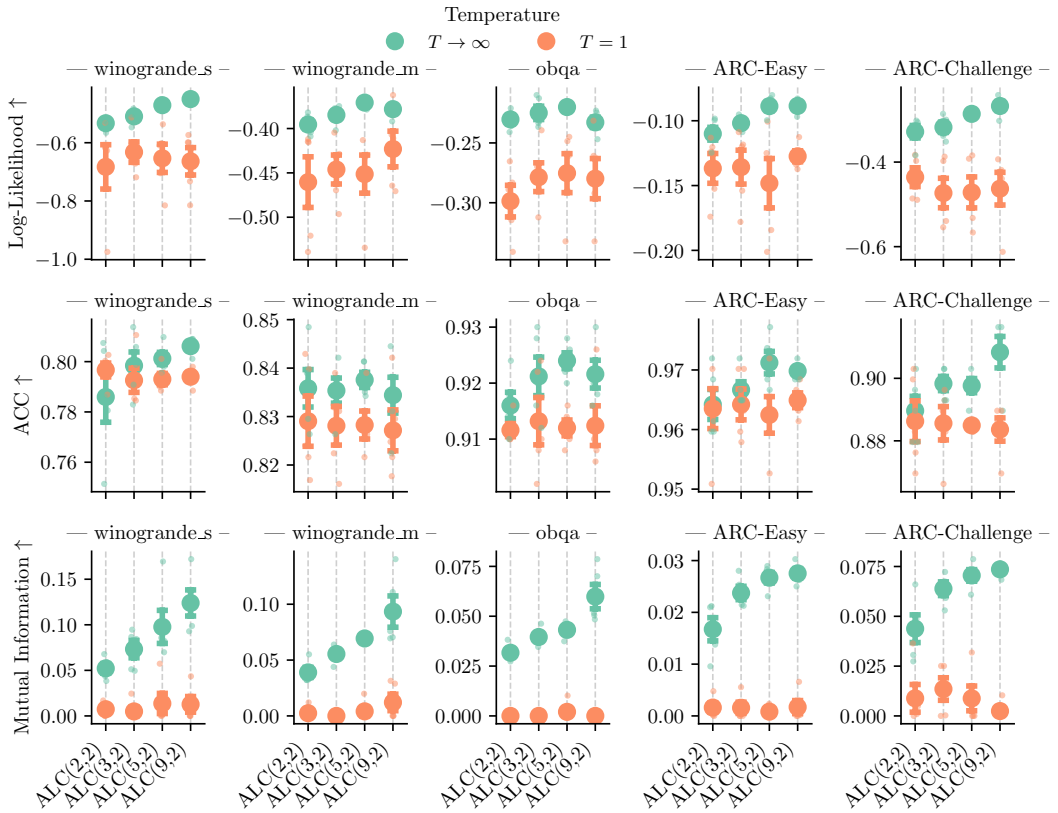


Figure 6: Effect of temperature scaling on Bayesian model averaging. Performance comparison of standard posterior weighting ($T = 1$, orange) versus uniform averaging ($T \to \infty$, green) across five datasets (columns). The rows display test Log-Likelihood (↑), Accuracy (ACC ↑), and Mutual Information (↑) for varying Anchored LoRA-Curve (ALC) complexities. Solid points with error bars indicate the mean and standard error, while faded dots represent individual runs.

C Performance comparison

Table 1 compares MAP, BLoB [28], ScalaBL [22], and DE [15] against our curve parameterizations. We reproduced the BLoB and ScalaBL baselines using the exact framework released by Samplawski et al. [22] across seeds $(0, 1, \ldots, 7)$. The DE and MAP results were extracted directly within our primary ALC experiments, as $\mathrm{ALC}(N, m)$ intrinsically encapsulates both $\mathrm{DE}(N)$ anchors and N individual MAP solutions.

Table 1: Performance comparison of Bayesian Model Averaging (BMA) strategies. Evaluation of Accuracy (ACC ↑), Expected Calibration Error (ECE ↓), and Log-Likelihood (LL ↑) across six datasets. N denotes the number of anchor points, and m represents the curve segment degree ($m = 0$: discrete/linear, $m > 0$: continuous curve). ALC (Anchored LoRA-Curve) uses frozen anchors from pretrained MAP solutions, while FLC (Free LoRA-Curve) jointly optimizes all control points. FGE(17) ensembles 17 snapshots of a single training run; FGE is not available for BoolQ. Results report the mean $\pm$ 2σ across available seeds. The best metric per dataset is **bolded**, and models whose mean $\pm$ 2σ interval contains the best result are underlined. Our continuous curve approaches (ALC and FLC) perform competitively with discrete deep ensembles (DE) in accuracy and log-likelihood and attain the highest MI, while the VI baselines are often better or on par in ECE.

Metric	Model	N	m	ARC-Challenge	ARC-Easy	obqa	winogrande_m	winogrande_s	boolq
ACC ↑	MAP	1	0	0.889 ± 0.021	0.962 ± 0.009	0.910 ± 0.016	0.820 ± 0.018	0.779 ± 0.030	0.892 ± 0.009
	SWA	—	—	0.898 ± 0.010	0.965 ± 0.012	0.912 ± 0.016	0.821 ± 0.010	0.796 ± 0.016	0.875 ± 0.003
	FGE(17)	—	—	0.896 ± 0.018	0.965 ± 0.007	0.917 ± 0.010	0.834 ± 0.013	0.797 ± 0.008	-
	BLoB	—	—	0.907 ± 0.010	0.965 ± 0.006	0.917 ± 0.013	0.828 ± 0.011	0.788 ± 0.012	0.891 ± 0.004
	ScalaBL	—	—	0.902 ± 0.012	0.963 ± 0.003	0.912 ± 0.009	0.817 ± 0.007	0.787 ± 0.009	0.883 ± 0.005
	Lin(3)	3	0	0.893 ± 0.021	0.966 ± 0.005	0.914 ± 0.013	0.823 ± 0.009	0.773 ± 0.040	0.894 ± 0.009
	DE(3)	3	0	0.897 ± 0.016	0.964 ± 0.005	0.920 ± 0.012	0.835 ± 0.018	0.787 ± 0.029	0.895 ± 0.006
	DE(9)	9	0	0.902 ± 0.011	0.970 ± 0.005	0.923 ± 0.011	**0.838 ± 0.013**	0.797 ± 0.016	0.897 ± 0.003
	ALC(2,1)	2	1	0.896 ± 0.018	0.965 ± 0.006	0.919 ± 0.008	0.833 ± 0.018	0.790 ± 0.015	**0.900 ± 0.005**
	ALC(3,2)	3	2	0.898 ± 0.010	0.967 ± 0.006	0.921 ± 0.015	0.835 ± 0.012	0.798 ± 0.024	0.899 ± 0.005
	FLC(3,2)	3	2	0.898 ± 0.015	0.966 ± 0.009	0.922 ± 0.006	0.833 ± 0.017	0.796 ± 0.019	-
	ALC(5,2)	5	2	0.898 ± 0.010	**0.971 ± 0.008**	**0.924 ± 0.006**	0.838 ± 0.007	0.801 ± 0.011	0.898 ± 0.006
	FLC(5,2)	5	2	0.903 ± 0.023	0.971 ± 0.005	0.923 ± 0.008	0.835 ± 0.012	0.801 ± 0.017	-
	ALC(9,2)	9	2	**0.908 ± 0.022**	0.970 ± 0.003	0.922 ± 0.011	0.834 ± 0.017	**0.806 ± 0.007**	0.895 ± 0.008
	FLC(9,2)	9	2	0.906 ± 0.013	0.970 ± 0.002	0.918 ± 0.009	0.831 ± 0.005	0.802 ± 0.013	0.876 ± 0.001
ECE ↓	MAP	1	0	0.068 ± 0.035	0.024 ± 0.012	0.034 ± 0.018	0.063 ± 0.033	0.103 ± 0.044	0.023 ± 0.017
	SWA	—	—	0.073 ± 0.019	0.021 ± 0.006	0.029 ± 0.010	0.066 ± 0.009	0.141 ± 0.017	0.024 ± 0.004
	FGE(17)	—	—	0.069 ± 0.026	0.022 ± 0.007	0.033 ± 0.022	0.083 ± 0.012	0.126 ± 0.030	-
	BLoB	—	—	**0.034 ± 0.014**	0.015 ± 0.008	**0.022 ± 0.009**	0.043 ± 0.009	0.072 ± 0.012	**0.015 ± 0.007**
	ScalaBL	—	—	0.044 ± 0.006	0.018 ± 0.004	0.024 ± 0.018	0.037 ± 0.010	0.086 ± 0.011	0.022 ± 0.003
	Lin(3)	3	0	0.047 ± 0.021	0.023 ± 0.011	0.022 ± 0.011	0.044 ± 0.009	0.080 ± 0.006	0.034 ± 0.049
	DE(3)	3	0	0.050 ± 0.023	0.018 ± 0.007	0.029 ± 0.007	0.034 ± 0.017	0.078 ± 0.016	0.016 ± 0.004
	DE(9)	9	0	0.046 ± 0.015	0.017 ± 0.006	0.025 ± 0.011	**0.028 ± 0.006**	0.069 ± 0.008	0.017 ± 0.006
	ALC(2,1)	2	1	0.061 ± 0.014	0.021 ± 0.005	0.028 ± 0.019	0.047 ± 0.016	0.088 ± 0.041	0.025 ± 0.016
	ALC(3,2)	3	2	0.056 ± 0.021	0.014 ± 0.009	0.029 ± 0.014	0.032 ± 0.023	0.077 ± 0.035	0.017 ± 0.008
	FLC(3,2)	3	2	0.060 ± 0.019	0.015 ± 0.007	0.026 ± 0.009	0.049 ± 0.037	0.078 ± 0.025	-
	ALC(5,2)	5	2	0.043 ± 0.019	0.016 ± 0.008	0.024 ± 0.015	0.033 ± 0.016	0.054 ± 0.025	0.022 ± 0.022
	FLC(5,2)	5	2	0.051 ± 0.010	0.015 ± 0.009	0.027 ± 0.015	0.035 ± 0.011	0.072 ± 0.046	-
	ALC(9,2)	9	2	0.041 ± 0.016	**0.012 ± 0.005**	0.035 ± 0.017	0.030 ± 0.020	**0.039 ± 0.026**	0.038 ± 0.010
	FLC(9,2)	9	2	0.038 ± 0.018	0.018 ± 0.008	0.031 ± 0.021	0.033 ± 0.018	0.048 ± 0.032	0.022 ± 0.004
LL ↑	MAP	1	0	-0.393 ± 0.180	-0.131 ± 0.050	-0.258 ± 0.045	-0.432 ± 0.039	-0.583 ± 0.148	-0.273 ± 0.020
	SWA	—	—	-0.352 ± 0.041	-0.114 ± 0.017	-0.261 ± 0.017	-0.436 ± 0.016	-0.701 ± 0.057	-0.308 ± 0.009
	FGE(17)	—	—	-0.367 ± 0.120	-0.113 ± 0.012	-0.241 ± 0.037	-0.439 ± 0.023	-0.646 ± 0.169	-
	BLoB	—	—	-0.278 ± 0.013	-0.095 ± 0.007	**-0.201 ± 0.015**	-0.377 ± 0.018	-0.485 ± 0.016	**-0.225 ± 0.010**
	ScalaBL	—	—	-0.295 ± 0.017	-0.109 ± 0.006	-0.224 ± 0.008	-0.393 ± 0.013	-0.501 ± 0.015	-0.237 ± 0.007
	Lin(3)	3	0	-0.297 ± 0.037	-0.111 ± 0.014	-0.243 ± 0.022	-0.408 ± 0.025	-0.536 ± 0.041	-0.273 ± 0.035
	DE(3)	3	0	-0.313 ± 0.055	-0.108 ± 0.021	-0.225 ± 0.018	-0.384 ± 0.022	-0.520 ± 0.050	-0.259 ± 0.011
	DE(9)	9	0	-0.287 ± 0.023	-0.098 ± 0.013	-0.218 ± 0.013	-0.373 ± 0.020	-0.493 ± 0.021	-0.253 ± 0.006
	ALC(2,1)	2	1	-0.332 ± 0.045	-0.115 ± 0.021	-0.230 ± 0.014	-0.394 ± 0.018	-0.542 ± 0.073	-0.257 ± 0.012
	ALC(3,2)	3	2	-0.318 ± 0.054	-0.102 ± 0.010	-0.225 ± 0.025	-0.385 ± 0.021	-0.510 ± 0.047	-0.251 ± 0.008
	FLC(3,2)	3	2	-0.339 ± 0.058	-0.102 ± 0.020	-0.231 ± 0.015	-0.402 ± 0.013	-0.505 ± 0.042	-
	ALC(5,2)	5	2	-0.286 ± 0.017	-0.089 ± 0.015	-0.220 ± 0.006	**-0.371 ± 0.010**	-0.471 ± 0.030	-0.258 ± 0.022
	FLC(5,2)	5	2	-0.282 ± 0.017	-0.093 ± 0.014	-0.229 ± 0.013	-0.375 ± 0.013	-0.498 ± 0.094	-
	ALC(9,2)	9	2	-0.267 ± 0.046	**-0.089 ± 0.011**	-0.233 ± 0.019	-0.378 ± 0.016	**-0.450 ± 0.026**	-0.273 ± 0.006
	FLC(9,2)	9	2	**-0.264 ± 0.017**	-0.097 ± 0.004	-0.236 ± 0.021	-0.389 ± 0.012	-0.459 ± 0.024	-0.298 ± 0.002
MI ↑	MAP	1	0	0.000 ± 0.000	0.000 ± 0.000	0.000 ± 0.000	0.000 ± 0.000	0.000 ± 0.000	0.000 ± 0.000
	SWA	—	—	0.000 ± 0.000	0.000 ± 0.000	0.000 ± 0.000	0.000 ± 0.000	0.000 ± 0.000	0.000 ± 0.000
	FGE(17)	—	—	0.034 ± 0.004	0.012 ± 0.003	0.021 ± 0.001	0.038 ± 0.008	0.050 ± 0.011	-
	Lin(3)	3	0	0.027 ± 0.009	0.008 ± 0.005	0.017 ± 0.003	0.026 ± 0.005	0.023 ± 0.021	0.027 ± 0.024
	DE(3)	3	0	0.035 ± 0.017	0.013 ± 0.007	0.023 ± 0.009	0.031 ± 0.009	0.030 ± 0.031	0.013 ± 0.004
	DE(9)	9	0	0.051 ± 0.012	0.018 ± 0.006	0.033 ± 0.003	0.047 ± 0.009	0.047 ± 0.024	0.016 ± 0.003
	ALC(2,1)	2	1	0.040 ± 0.012	0.015 ± 0.010	0.026 ± 0.017	0.034 ± 0.022	0.049 ± 0.027	0.017 ± 0.005
	ALC(3,2)	3	2	0.064 ± 0.015	0.024 ± 0.006	0.040 ± 0.009	0.055 ± 0.015	0.073 ± 0.043	0.020 ± 0.004
	FLC(3,2)	3	2	0.060 ± 0.013	0.021 ± 0.004	0.038 ± 0.011	0.056 ± 0.016	0.070 ± 0.021	-
	ALC(5,2)	5	2	0.071 ± 0.013	0.027 ± 0.005	0.043 ± 0.008	0.069 ± 0.007	0.098 ± 0.081	0.037 ± 0.038
	FLC(5,2)	5	2	0.061 ± 0.012	0.022 ± 0.002	0.045 ± 0.018	0.073 ± 0.055	0.077 ± 0.030	-
	ALC(9,2)	9	2	**0.074 ± 0.007**	**0.028 ± 0.004**	**0.060 ± 0.027**	**0.093 ± 0.063**	**0.124 ± 0.064**	**0.073 ± 0.008**
	FLC(9,2)	9	2	0.065 ± 0.013	0.027 ± 0.006	0.051 ± 0.023	0.087 ± 0.053	0.107 ± 0.049	0.015 ± 0.003

D Additional results

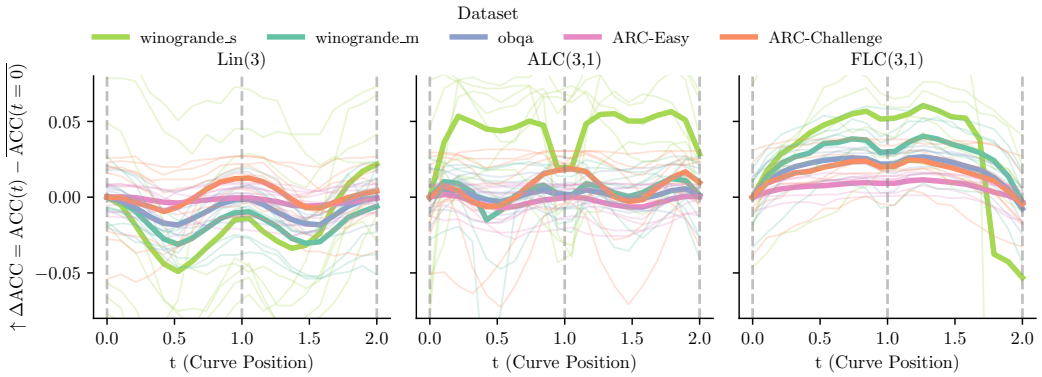


Figure 7: Same as Figure 2, but evaluating the change in accuracy on y-axis

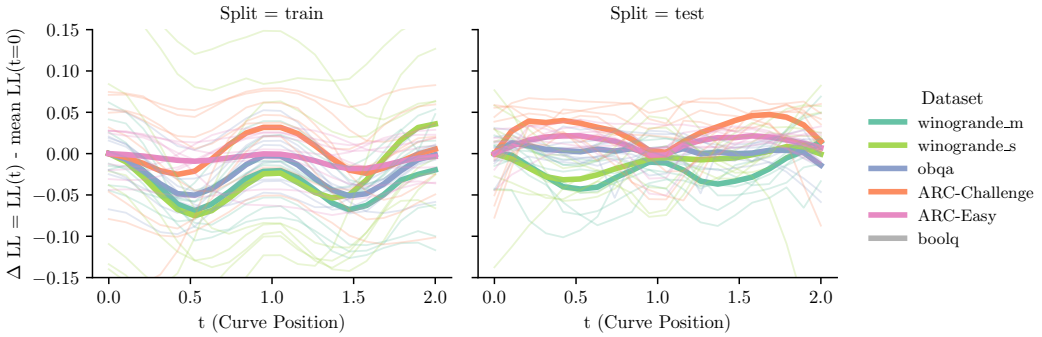


Figure 8: Linear mode connectivity for train and test splits. The left panel depicts the relative change in log-likelihood (Δ LL) for a Lin(3) model, identical to the setup in Figure 2. The right panel extends this analysis to the test data, illustrating how the training loss barriers (dips at $t = 0.5$ and $t = 1.5$) manifest differently on the test set across various datasets.

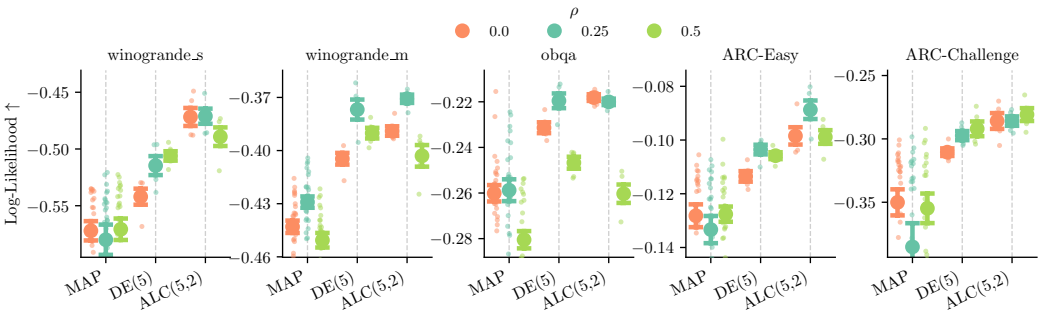


Figure 9: Same comparison as in Figure 5a, but evaluated on the Winogrande S/M, OBQA, and ARC-Challenge/Easy datasets, which are displayed across different columns.

D.1 LoRA rank

To assess the influence of the LoRA rank, we repeat the DE and ALC experiments with rank $r = 40$, so that a single member consumes as much memory as DE(5) with $r = 8$. Increasing the rank from 8 to 40 yields no systematic improvement for either method (Figure 13). Moreover, DE(5) with

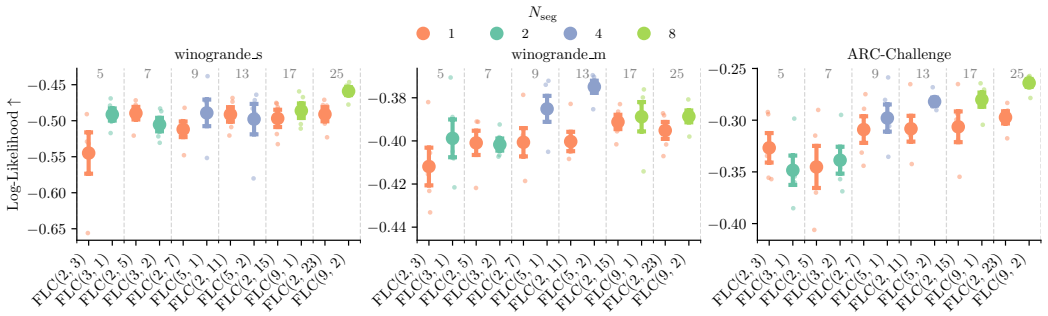


Figure 10: Log-likelihood comparison of multi-segment vs. single-segment Bézier curves. Test data log-likelihood (↑) is evaluated across three datasets (panels). Models are grouped by the total number of control points, indicated by the gray numbers at the top of each section. Color coding denotes the number of segments (N_{seg}), with the orange baseline representing the single-segment approach. For a fixed control point budget, multi-segment curves consistently match or outperform their single-segment counterparts.

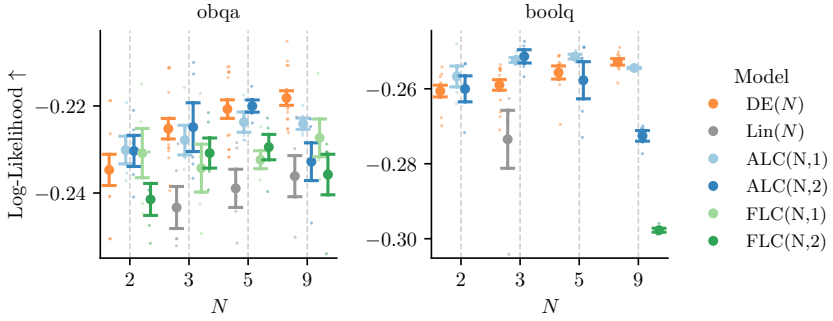


Figure 11: Extended test log-likelihood comparison. Same as Figure 4, but extended to include the `obqa` and `boolq` datasets. The panels compare our continuous LoRA-Curve parameterizations against discrete Deep Ensembles (DE, orange) using test log-likelihood (↑) evaluated for varying numbers of anchors (N, x-axis). Solid points with error bars denote the mean and standard error across 5 random seeds, shown as faded dots.

$r = 8$, which requires the same adapter memory as a single rank-40 adapter, outperforms ALC(2,1) with $r = 40$ on all datasets, and ALC(5,2) with $r = 8$ improves further on four of five datasets, i.e., allocating memory to multiple modes or a richer curve is more beneficial than increasing the rank.

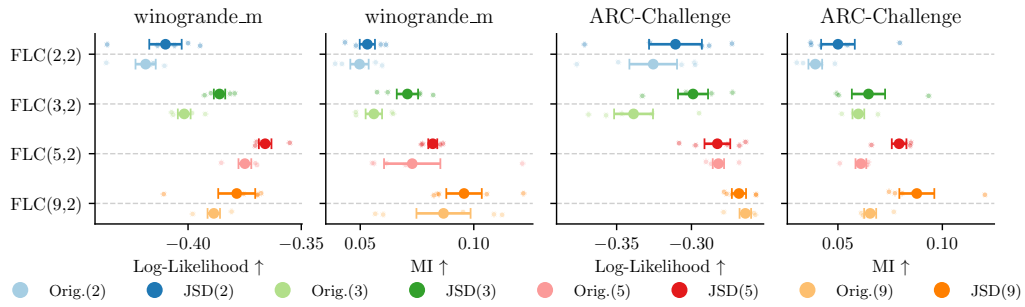


Figure 12: Functional diversity via JSD regularization. Comparison of Log-Likelihood (↑) and mutual information (MI) (↑) across two datasets (panels) for varying FLC model complexities (y-axis). Colors distinguish between the unregularized baseline (Orig., lighter shades) and the JSD-regularized models (JSD, darker shades). Solid points with error bars indicate the mean and standard error, while faded dots represent individual runs. The addition of the repulsive JSD regularizer significantly improves MI while maintaining comparable or slightly improved Log-Likelihood.

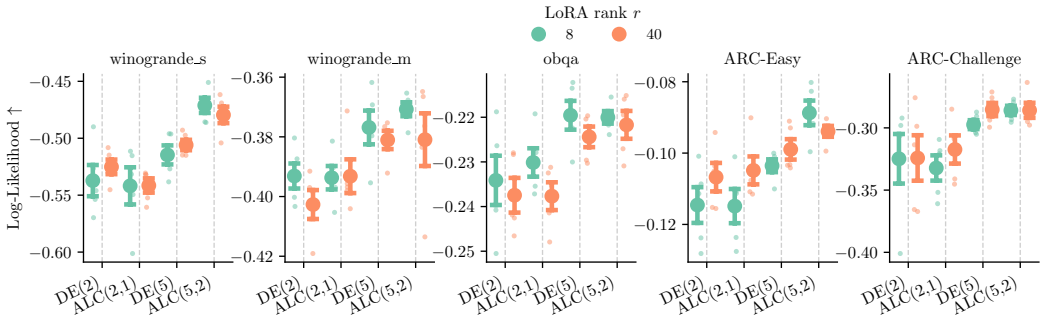


Figure 13: Test log-likelihood (↑) for LoRA rank $r = 8$ and $r = 40$. Solid points with error bars denote the mean and standard error across 5 random seeds, shown as faded dots. $N = 9$ is omitted as it exceeded the GPU memory at $r = 40$, and BoolQ was not evaluated at $r = 40$.

D.2 Additional smoothness on OOD examples

To further rigorously evaluate predictive smoothness on highly uncertain data, we constructed specialized out-of-distribution (OOD) examples in which exactly one, multiple, or no candidate classes are correct (cf. Section D.2.1). As shown in Figure 14, we consistently observe smooth transitions between class probabilities despite these distributional shifts. Notably, even when no valid class is provided, the predictive distribution can still transition smoothly across all candidate labels (e.g., OOD Example 6). However, we note that all evaluated methods occasionally exhibit failure modes on these extreme edge cases, as demonstrated by the breakdown in OOD Example 4 (cf. Figure 14).

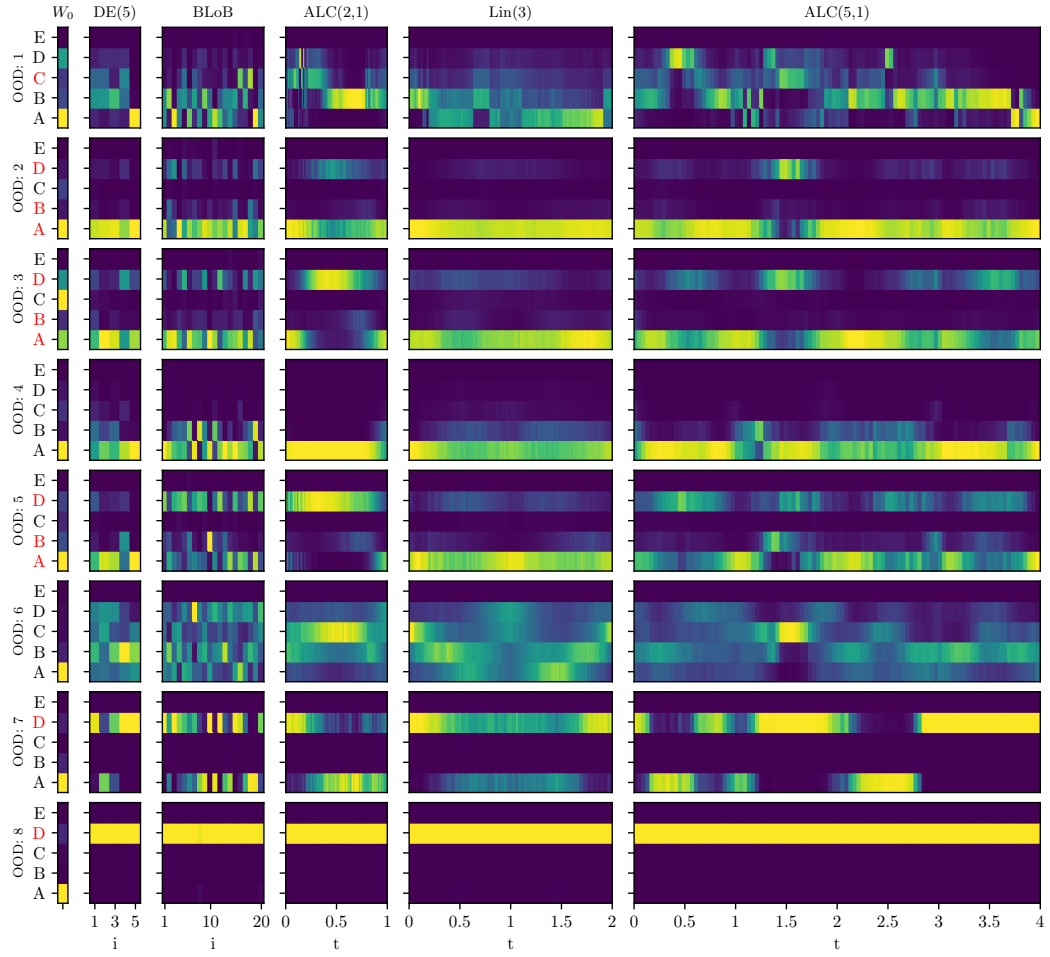


Figure 14: The class probability evolution for OOD examples using our fine-tuned LoRA model on the ARC-Challenge data. Predicted probabilities (linear color-coded from dark blue for 0 to yellow for 1) across five class labels (A–E) for eight OOD test examples (rows). The red class on the y-axis depicts the correct ground-truth label for each respective example. Facet columns compare various approaches: the base Qwen 2.5 model without pretraining (W_0), a 5-member Deep Ensemble (DE(5)), the BLoB variational competitor, linear interpolation between 3 anchors (Lin(3)), and our continuous Anchored LoRA-Curve methods (ALC(2,1) and ALC(5,1)). The x-axis represents either the discrete model index (i) or the continuous curve parameter (t).

D.2.1 OOD examples

To rigorously evaluate predictive behavior under distributional shift, we generated a custom set of out-of-distribution (OOD) examples structurally aligned with the ARC-Challenge dataset. Using Gemini, we carefully crafted questions that deliberately violate the dataset's standard single-answer

assumption by requiring multiple correct labels (e.g., assigning ground-truth classes `[0, 1, 3]`). This multi-label extension provides a realistic simulation of OOD scenarios, forcing the model to navigate edge cases where standard predictive confidence typically breaks down.

```
- id: 1
  prompt:
    Return the label of the correct answer for the question below.

    Question: What is the value of x in the equation 3x = 12?
    Choices:
    2) A
    3) B
    4) C
    5) D
    Answer:
  label: 2

- id: 2
  prompt:
    Return the label of the correct answer for the question below.

    Question: Which of the following are renewable energy sources?
    Choices:
    solar power) A
    wind power) B
    coal) C
    hydroelectric power) D
    Answer:
  label: [0, 1, 3]

- id: 3
  prompt:
    Return the label of the correct answer for the question below.

    Question: Which of the following statements correctly describe properties
    of democratic systems?
    Choices:
    citizens can vote in elections) A
    laws apply equally to all citizens) B
    power is inherited through family lineage) C
    governments are accountable to the public) D
    Answer:
  label: [0, 1, 3]

- id: 4
  prompt:
    Return the label of the correct answer for the question below.

    Question: What caused the event described above?
    Choices:
    friction) A
    gravity) B
    magnetism) C
    electricity) D
    Answer:
  label: []

- id: 5
```

```
  prompt:
    Return the label of the correct answer for the question below.

    Question: Which of the following are mammals?
    Choices:
    whale) A
    bat) B
    lizard) C
    human) D
    Answer:
  label: [0, 1, 3]

- id: 6
  prompt:
    Return the label of the correct answer for the question below.

    Question: Which answer choice is correct for this question?
    Choices:
    this one) A
    this one) B
    this one) C
    this one) D
    Answer:
  label: []

- id: 7
  prompt:
    Return the label of the correct answer for the question below.

    Question: A square has four equal sides and no equal sides. What shape is it?
    Choices:
    square) A
    circle) B
    triangle) C
    none of the above) D
    Answer:
  label: 3

- id: 8
  prompt:
    Return the label of the correct answer for the question below.

    Question: Which of the following has the highest moral responsibility?
    Choices:
    a rock) A
    a tree) B
    a river) C
    a conscious human) D
    Answer:
  label: 3
```

E Discarded approach

E.1 Repulsive regularization

Early in the development of the ALC framework, we hypothesized that enforcing geometric diversity directly in the parameter space would prevent mode collapse. To this end, we introduced a repulsive regularizer that penalized the pairwise cosine similarity between the curve's control points, ϑ_i. For N control points, this regularizer was defined as the sum of squared cosine similarities over all distinct pairs:

$$\mathcal{R}_{\text{repulsive}} = \sum_{1 \le i < j \le N_{\text{cp}}} \left(\frac{\vartheta_i \cdot \vartheta_j}{\|\vartheta_i\| \|\vartheta_j\|} \right)^2.$$

While this explicitly forced the control points to disperse across the LoRA space, it yielded inconsistent downstream performance. Crucially, geometric distance in weight space did not strongly correlate with functional diversity in the model's predictive distributions. Instead, strong repulsive forces often pushed the control points into higher-loss regions, causing the trajectory to deviate from the optimal low-loss valley.

Consequently, this weight-space regularization was discarded in favor of output-space regularization (such as the JSD regularizer described in Section 3.4), which directly encourages diversity in the predictive distributions without unnecessarily distorting the underlying geometric path.

E.2 JSD regularization with corrupted input

Further, we explored a variant of the JSD regularizer that maximizes functional diversity by explicitly enforcing predictive disagreement on out-of-distribution (OOD) inputs. Rather than evaluating the regularizer on clean training data, we computed the Jensen-Shannon divergence on dynamically corrupted inputs $\tilde{x}$. We evaluated two corruption strategies: replacing a random subset of input tokens with uniform random tokens from the vocabulary, and dynamically dropping a fraction of attention-masked tokens. For both variants, the loss was computed as:

$$\mathcal{L} = \frac{1}{2}\big(\mathcal{L}_{\text{CE}}(t_1) + \mathcal{L}_{\text{CE}}(t_2)\big) + \lambda_{\text{JSD}} \max(\tau_{\text{JSD}} - D_{\text{JSD}}(\tilde{p_{t_1}} \parallel \tilde{p_{t_2}}), 0),$$

where $t_2 = (t + t_{\max}/2) \pmod{t_{\max}}$ ensures maximal separation along the curve, and $\tilde{p_t} := \text{softmax}(f(\tilde{x}; W_\vartheta(t)))$ denotes the predictive distribution on the corrupted input.

The intuition was that forcing disagreement on OOD inputs would encourage the curve to explore a broader range of functional regions. Empirically, this formulation successfully achieved its direct objective: the JSD measured on the corrupted inputs $\tilde{x}$ increased substantially during training. However, this synthetic functional diversity did not translate to significant improvements in generalization performance. Because the severely distorted inputs $\tilde{x}$ lay too far outside the true data manifold.

Furthermore, this approach introduced a severe computational bottleneck. Evaluating the OOD disagreement alongside the standard loss necessitated four forward passes per gradient update step. This pushed memory requirements to the limit, restricting the feasibility of training exclusively on high-capacity hardware (e.g., H100 GPUs with 94GB VRAM).

Consequently, we reverted to the clean-data JSD regularizer (Section 3.4), which directly enforces diversity on the task-relevant input distribution without the prohibitive variance or excessive memory overhead of synthetic OOD corruption. Nevertheless, explicit OOD disagreement remains a promising future direction for architectures fundamentally designed to leverage heavily augmented views, such as contrastive learning frameworks.

F Hyperparameter selection

We largely align our core hyperparameters with those established by BLoB [28] and ScalaBL [22]. However, to ensure reliable convergence of the curve, we extend the training budget to 10,000 update steps, utilizing a one-cycle learning rate schedule peaking at 10^{-4} and selecting the checkpoint with the best validation log-likelihood on a 10% validation split. Crucially, this extended and regularized optimization regime explains why our MAP and DE baselines achieve significantly stronger performance than those previously reported by ScalaBL.

Furthermore, to accurately approximate the continuous trajectory, we dynamically scale the number of equidistant grid points M based on the curve's complexity by setting $M = 2N_{\text{cp}} - 1$. This formulation guarantees sufficient evaluation density, even for highly segmented curves, by explicitly placing one grid point at every control point—capturing its peak influence on the path—and one grid point midway between each adjacent pair.

For FGE, we start from a MAP solution trained for 5,000 steps and collect 5, 9, or 17 snapshots during 2,000 additional steps using a cyclical learning rate between 10^{-4} and 10^{-5}; the snapshots are averaged uniformly in prediction space. For SWA, we train a single LoRA model for 2,000 steps and average all iterates in weight space after the first 10% of training. FGE is not reported for BoolQ, since its cyclical phase following the MAP warmup exceeded the memory of our 48 GB GPUs at BoolQ's input length of 525 tokens.

All experiments were logged with Weights & Biases (W&B). Hyperparameters were explored using W&B sweeps. The hyperparameter search space and fixed settings are detailed in the following:

Table 2: Hyperparameters varied in the W&B project and their methodological role.

Hyperparameter	**Observed Values**	**Role in Method**
`num_curve_segment`	`1, 2, 4, 8`	Controls the number of segments N_{seg}. Higher values allow the curve to navigate more complex landscapes.
`SegDeg`	**1, 2, 3,** `4, 6, 8, 12, 16, 24`	Degree of a single Bézier segment (similar to $m + 1$).
`Pretraining`	[True, False]	If True it uses ALC otherwise FLC
`rng_seed`	`1, 2, 20, 400, 8000`	Random seeds for initialization and sampling.

Table 3: Fixed hyperparameters and configurations maintained across all experiments. JSD hyperparameters only for JSD experiments. These parameters were also optimized through a previous small grid analysis on the *Winogrande s* validation dataset.

Category	**Hyperparameter**	**Value**	**Description**
LoRA	r	8	Dimensionality of the low-rank updates.
	α	16	Alpha scaling factor applied to the LoRA matrices.
	`lora_layers`	value, query, lm_head	Layer names of Qwen2.5 which are adapted.
Optimization	`optimizer`	AdamW	Algorithm utilized for gradient descent.
	`weight_decay`	0.001	Regularization penalty applied to weights.
	`val_percentage`	10% (BoolQ 2.5%)	Percentage of validation date out of original train data.
LR Schedule	`type`	One-cycle	Learning rate schedule policy.
	`steps`	$10,000$	Total duration of the training schedule.
	`peak_value`	0.0001	peak learning rate.
	`pct_start`	0.12	Percentage of steps spent increasing the learning rate.
	`div_factor`	300	Initial learning rate division factor.
Training	`batch_size`	4	Number of samples processed per training step.
Flat-LoRA	ρ	0.25	Magnitude of the perturbation noise.
	`rho_sched_freq`	50	Frequency for noise scheduling.
JSD	λ_{JSD}	0.2	Magnitude of the JSD regularization.
	τ_{JSD}	0.05	Cut-off value for JSD

G Computational environment and performance metrics

This section details the hardware, software, and runtime performance characteristics of our LoRA-Curve experiments. Code to reproduce all experiments, including the sweep configurations, is available at `https://github.com/doldd/Time_Transformer`. A lightweight standalone library extracted from it, intended as a starting point for building on LoRA-Curve, is available at `https://github.com/doldd/bayes_sub_inf`.

G.1 Hardware and software specifications

All experiments were conducted on high-performance compute nodes equipped with the following NVIDIA A100 (40GB VRAM) and NVIDIA L40S (48GB VRAM) GPU nodes.

The software stack utilized for the implementation and orchestration of experiments included:

- **Language:** Python 3.10.18.
- **Frameworks:** JAX 0.5.0, Flax 0.10.0, and PyTorch (for weight loading and data processing).
- **Tracking:** Weights & Biases (W&B) for experiment tracking and metric logging.

G.2 Model and training statistics

The base model used for all experiments is Qwen2.5 7B [30], which contains approximately 7.65 billion parameters. When fine-tuned using LoRA parameters in float32 precision and original Qwen2.5 W_0 in bfloat16 precision, the total parameter memory footprint is approximately 14.6 GB. The number of trainable parameters varies with the curve resolution:

- **Anchored LoRA-Curve (ALC):** Since the anchors are frozen, only the intermediate handle points are trainable. For a rank $r = 8$ configuration, trainable parameters range from 11M to 64M depending on the number of anchors N and handles per segment m.
- **Training Budget:** Full curves (FLC and ALC) are optimized for 10,000 steps. In ALC, the individual anchor points (corresponding to DE members) are independently pretrained for half the number of epochs (approximately 5,000 steps each) at half the peak learning rate.

G.3 FLOP analysis

The dominant cost of an update step is the forward and backward pass. Since the curve point $\zeta_\vartheta(t)$ is only a weighted linear combination of the control points, FLC requires a single forward and backward pass per step, independent of N and m. Table 4 confirms this empirically: even for FLC(9,2), the FLOPs per step exceed those of a MAP update by only 0.0022%. With J update steps per model, DE(N) thus costs $\mathcal{O}(NJ)$, FLC $\mathcal{O}(2J)$ (as it runs for twice the steps of a single model), and ALC $\mathcal{O}(NJ + 2J)$, i.e., the anchor training of DE(N) plus one curve optimization. At inference, DE(N) requires N forward passes, whereas ALC and FLC evaluate $M = 2N_{\text{cp}} - 1$ grid points.

Table 4: FLOPs of a single training step (forward and backward pass) for MAP and FLC(N,2) with batch size 4 and sequence length 256 on Qwen2.5 7B.

Method	N	FLOPs/step	vs. MAP
MAP	–	3.134449×10^{13}	$1.000000\times$
FLC	2	3.134460×10^{13}	$1.000003\times$
FLC	3	3.134464×10^{13}	$1.000005\times$
FLC	5	3.134477×10^{13}	$1.000009\times$
FLC	9	3.134517×10^{13}	$1.000022\times$

G.4 Training and inference cost

We measure training cost in optimizer steps, which is independent of the hardware; wall-clock times are given in Table 6. Since all methods share the same per-step cost up to a negligible overhead

(Table 4), steps are proportional to FLOPs. We report both the step budget, i.e., the steps actually trained, and the steps until the checkpoint selected by validation log-likelihood (Figure 15 and Table 5). Step budgets are hyperparameters and were not tuned for compute efficiency, so Figure 15 should be read for the budgets used here. SWA averages all iterates without checkpoint selection and was therefore given a shorter budget of 2,000 steps.

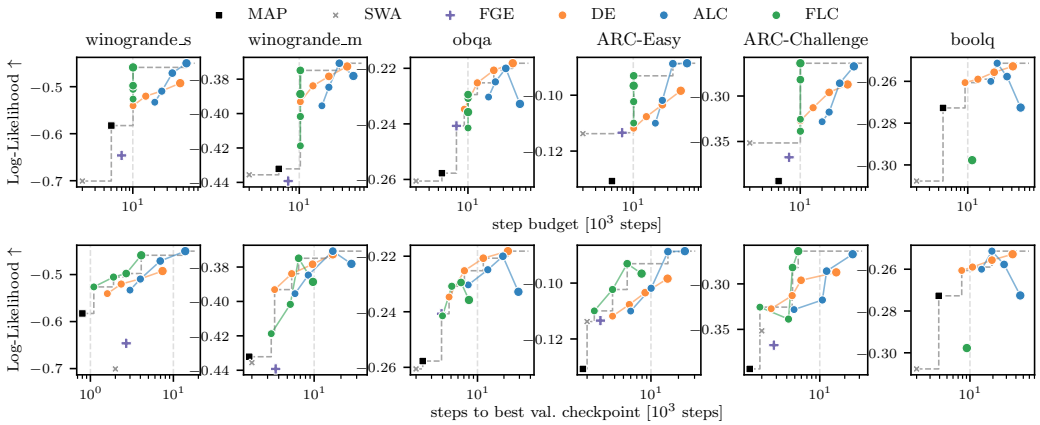


Figure 15: Training cost vs. test log-likelihood (↑). Top: step budget; bottom: steps until the best validation checkpoint (log scale; step counts are medians and log-likelihoods means over seeds). Lines connect configurations of the same method with increasing N (marker size); all curve configurations use $m = 2$. The dashed line marks the Pareto front.

Table 5: Training cost in 10^3 optimizer steps (median across the six datasets; range in brackets) and forward passes per test example. All curve configurations use $m = 2$.

Method	Step budget	Steps to best val. checkpoint	Forward passes
MAP	5.0	1.8 [0.8–3.9]	1
SWA	2.0	2.0	1
FGE(17)	7.0	2.8 [2.7–3.9]	17
DE(2)	10.1	3.7 [1.6–7.8]	2
DE(3)	15.1	5.7 [2.4–10.8]	3
DE(5)	25.2	9.1 [4.0–19.4]	5
DE(9)	45.3	16.0 [7.4–36.2]	9
ALC(2,2)	20.2	6.0 [3.0–14.2]	7
ALC(3,2)	25.2	10.4 [4.0–19.4]	13
ALC(5,2)	35.2	15.8 [7.0–28.0]	25
ALC(9,2)	55.3	25.6 [13.9–45.6]	49
FLC(2,2)	10.1	2.4 [1.1–4.0]	7
FLC(3,2)	10.1	4.2 [1.9–5.4]	13
FLC(5,2)	10.1	5.5 [2.7–6.7]	25
FLC(9,2)	10.1	8.0 [4.1–9.7]	49

Inference cost At test time, DE(N) requires N forward passes per example, whereas ALC and FLC are evaluated at $M = 2N_{\text{cp}} - 1$ grid points (Section F), i.e., the inference cost grows with the number of control points (e.g., $M = 25$ for ALC(5,2) vs. 5 for DE(5)). This rule was chosen to densely cover the curve and was not tuned: for ALC(5,2), reducing M from 25 to 9 changes the test log-likelihood by at most 0.004 nats across all six datasets, while the 5 grid points at the anchors recover DE(5).

G.5 Runtime comparison

Across our W&B project, we logged over 980 runs, for which we report typical training times across different curve configurations in Table 6. The total compute investment for the results presented in this work is estimated at approximately 2500 GPU-hours.

Note that the FLC walltime decreases for simpler curves because we dynamically scale the number of equidistant grid points in our approximation based on the number of control points. Furthermore, while FLC theoretically avoids the $N\times$ sequential training overhead inherent to pretraining ALC anchors, the empirical speedup does not scale exactly by a factor of N. This discrepancy arises because the reported wall times include fixed JAX compilation and evaluation overheads, and because individual ALC anchors are only pretrained for 5,000 steps, whereas the curve optimization runs for the full 10,000 steps.

Table 6: Runtime for a single experiment for different LoRA-Curve configurations and datasets. Times are reported for 10,000 curve training steps (for ALC additionally including the anchor pretraining) on a single NVIDIA L40S GPU. The reported time includes training time, evaluation of test performance, and an additional evaluation using the training and validation data, which are only used for diagnostic purposes.

Method(N, m)	Dataset	**Time (h)**
ALC$(9, 2)$	BoolQ	18h 50m
FLC$(9, 2)$	BoolQ	12h 10m
ALC$(2, 2)$	BoolQ	6h 10m
ALC$(9, 2)$	OBQA	5h 50m
FLC$(9, 2)$	OBQA	3h 15m
ALC$(5, 2)$	OBQA	4h 50m
FLC$(5, 2)$	OBQA	2h 44m
ALC$(3, 2)$	OBQA	3h 23m
FLC$(3, 2)$	OBQA	2h 30m
ALC$(2, 2)$	OBQA	2h 55m
FLC$(2, 2)$	OBQA	2h 20m
ALC$(9, 2)$	Winogrande m	3h 40m
FLC$(9, 2)$	Winogrande m	1h 35m
ALC$(2, 2)$	Winogrande m	1h 39m
FLC$(2, 2)$	Winogrande m	1h 10m

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The claims made in the abstract and introduction regarding the construction and functional diversity of continuous low-loss valleys are directly supported by the theoretical propositions (e.g., Proposition 1) and the experimental results (e.g., Figures 3 and 4).

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Limitations regarding dimensionality scaling, and optimization gaps are explicitly discussed in Section 6.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [Yes]

Justification: The paper provides necessary assumptions and complete derivations for pathwise continuity and Lipschitz regularity directly following Propositions 1 and 2 in Section 4.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: All experiments are logged using Weights % Biases, and complete training details, including the number of steps, learning rate schedules, and a comprehensive hyperparameter breakdown, are provided in Section F.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: The paper provides an overview of the hyperparameters used in Section F to ensure reproducibility. Code to reproduce all experiments is available at https://github.com/doldd/Time_Transformer; a lightweight standalone library extracted from it for building on LoRA-Curve is available at https://github.com/doldd/bayes_sub_inf.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (https://neurips.cc/public/guides/CodeSubmissionPolicy) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (https://neurips.cc/public/guides/CodeSubmissionPolicy) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: Detailed training and test configurations, including data splits, optimizer settings, and hyperparameter selection via W&B sweeps, are specified in Section F.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: All experiments were repeated at least five times with different seeds to compute error bars.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.

- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: There is a dedicated section in the appendix about this cf. Section G.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The research conforms to the NeurIPS Code of Ethics, as it focuses on methodological improvements for uncertainty estimation using publicly available models and datasets.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [N/A]

    Justification: The paper focuses on the fundamental methodology of posterior geometry in parameter-efficient fine-tuning. A discussion of its broader societal implications is deferred to future work.

Guidelines:

- The answer [N/A] means that there is no societal impact of the work performed.
- If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: The study uses only established, publicly available models, such as Qwen 2.5, and standard open-source benchmarks, posing no additional risks of misuse beyond their existing releases.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: The creators of all utilized assets, including the Qwen 2.5 model and reasoning benchmarks, are properly credited through citations.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.

- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: The implementation of the *LoRA-Curve* framework is provided as a self-contained and documented anonymized zip file in the supplemental material, with further details in Section F.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: This study does not involve crowdsourcing or human subjects, as it relies entirely on automated evaluations using standard open-source benchmarks.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

Justification: This research does not involve human subjects or study participants, and thus did not require Institutional Review Board (IRB) approval or risk disclosures.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

Answer: [N/A]

Justification: The core method development in this research does not involve LLMs as an agent or algorithmic component.

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.