Variational Deep Learning via Implicit Regularization

Anonymous Author(s)
Affiliation
Address
email

Abstract

Modern deep learning models generalize remarkably well in-distribution, despite being overparametrized and trained with little to no *explicit* regularization. Instead, current theory credits *implicit* regularization imposed by the choice of architecture, hyperparameters and optimization procedure. However, deploying deep learning models out-of-distribution, in sequential decision-making tasks, or in safety-critical domains, necessitates reliable uncertainty quantification, not just a point estimate. The machinery of modern approximate inference — Bayesian deep learning — should answer the need for uncertainty quantification, but its effectiveness has been challenged by the associated computational burden and by difficulties in defining useful *explicit* inductive biases through priors. Instead, in this work we demonstrate, both theoretically and empirically, how to regularize a variational deep network *implicitly via the optimization procedure*, just as for standard deep learning. We fully characterize the inductive bias of (stochastic) gradient descent in the case of an overparametrized linear model as generalized variational inference and demonstrate the importance of the choice of parametrization. Finally, we show empirically that our approach achieves strong in- and out-of-distribution performance without tuning of additional hyperparameters and with minimal time and memory overhead over standard deep learning.

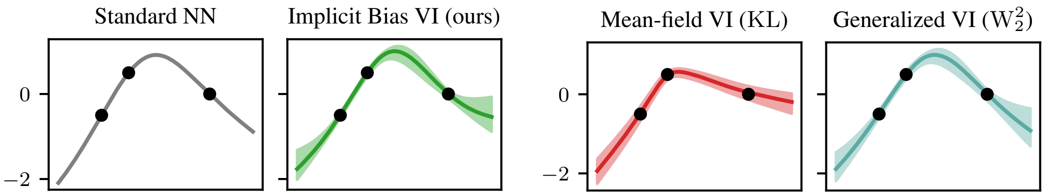
1 Introduction

The success of deep learning across many application domains is, on the surface, remarkable, given that deep neural networks are usually overparameterized and trained with little to no *explicit* regularization. The generalization properties observed in practice have been explained by *implicit* regularization instead, resulting from the choice of architecture [1], hyperparameters [2, 3], and optimizer [4–10]. Notably, the corresponding inductive biases often require no additional computation, in contrast to enforcing a desired inductive bias through explicit regularization.

In the last two decades, there has been an increasing focus on improving the reliability and robustness of deep learning models via (approximately) Bayesian approaches [11] to improve performance on out-of-distribution data [12], in continual learning [13] and sequential decision-making [14]. However, despite its promise, in practice, Bayesian deep learning can suffer from issues with prior elicitation [15], can be challenging to scale [16], and explicit regularization from the prior combined with approximate inference may result in pathological inductive biases and uncertainty [17–20].

In this work, we demonstrate both theoretically and empirically how to exploit the implicit bias of optimization for approximate inference in probabilistic neural networks, thus regularizing training implicitly rather than explicitly via the prior. This not only narrows the gap to how standard neural networks are trained, but also reduces the computational overhead of training compared to variational inference. More specifically, we propose to learn a variational distribution over the weights

Submitted to the NeurIPS 2025 Workshop on Structured Probabilistic Inference & Generative Modeling.



(a) *Implicit* regularization.

(b) *Explicit* regularization.

Figure 1: *Variational deep learning via implicit regularization.* Neural networks generalize well without explicit regularization due to implicit regularization from the architecture and optimization. We can exploit this implicit bias for variational deep learning, removing the computational overhead of explicit regularization and narrowing the gap to deep learning practice. As illustrated for a two-hidden layer MLP and proven rigorously for overparametrized linear models in Theorems 1 and 2, the implicit bias of (S)GD in variational networks (see (a)) can be understood as generalized variational inference with a 2-Wasserstein regularizer (see (b)). This differs from the standard ELBO objective with a KL divergence to the prior as used for example in mean-field VI (see (b)).

of a deep neural network by maximizing the *expected* log-likelihood in analogy to training via maximum likelihood in the standard case. However, in contrast to variational Bayes, there is *no explicit regularization* via a Kullback-Leibler divergence to the prior. Surprisingly, we show theoretically and empirically that training this way does not cause uncertainty to collapse away from the training data, if initialized and parametrized correctly. More so, for overparametrized linear models we rigorously characterize the implicit bias of SGD as generalized variational inference with a 2-Wasserstein regularizer penalizing deviations from the prior. Figure 1 illustrates our approach on a toy example.

Contributions In this work, we propose a new approach to uncertainty quantification in deep learning that exploits the implicit regularization of (stochastic) gradient descent. We precisely characterize this implicit bias for regression (Theorem 1) and binary classification (Theorem 2) in overparameterized linear models, generalizing results for non-probabilistic models and drawing a rigorous connection to generalized Bayesian inference. We also demonstrate the importance of the parametrization for the inductive bias and its impact on hyperparameter choice. In several benchmarks, we demonstrate competitive performance to state-of-the-art baselines for Bayesian deep learning, at minimal computational overhead compared to standard neural networks. Finally, we provide an open-source implementation of our approach in a standalone library: `inferno`.

2 Background

Given a training dataset $(\boldsymbol{X}, \boldsymbol{y}) = \{(\boldsymbol{x}_n, y_n)\}_{n=1}^N$ of input-output pairs, supervised learning seeks a model $f_{\boldsymbol{w}}(\boldsymbol{x})$ to predict the corresponding output $y(\boldsymbol{x})$ for a test input $\boldsymbol{x}$. The parameters $\boldsymbol{w} \in \mathbb{R}^P$ of the model are typically trained via empirical risk minimization, i.e.

$$\boldsymbol{w}_\star \in \arg\min_{\boldsymbol{w}} \ell_R(\boldsymbol{w}) \qquad \text{with} \qquad \ell_R(\boldsymbol{w}) = \ell(\boldsymbol{y}, f_{\boldsymbol{w}}(\boldsymbol{X})) + \lambda R(\boldsymbol{w}), \tag{1}$$

where the loss $\ell(\boldsymbol{y}, f_{\boldsymbol{w}}(\boldsymbol{X}))$ encourages fitting the training data and the regularizer $R(\boldsymbol{w})$, given some $\lambda > 0$, discourages overfitting, which can lead to poor generalization on test data.

Implicit Bias of Optimization One of the remarkable observations in deep learning is that training overparametrized models ($P > N$) with gradient descent without *explicit* regularization can nonetheless lead to good generalization [21] because the optimizer, initialization, and parametrization *implicitly* regularize the optimization problem $\arg\min_{\boldsymbol{w}} \ell(\boldsymbol{y}, f_{\boldsymbol{w}}(\boldsymbol{X}))$ [e.g. 4, 5, 7, 22, 23].

Variational Inference Bayesian inference quantifies uncertainty in the parameters, and consequently predictions, by the posterior distribution $p(\boldsymbol{w} \mid \boldsymbol{X}, \boldsymbol{y}) \propto p(\boldsymbol{y} \mid \boldsymbol{X}, \boldsymbol{w})p(\boldsymbol{w})$, which depends on the choice of a prior belief $p(\boldsymbol{w})$ and a likelihood $p(y \mid \boldsymbol{w})$. Approximating the posterior with $q_{\boldsymbol{\theta}} \approx p(\boldsymbol{w} \mid \boldsymbol{X}, \boldsymbol{y})$ by maximizing a lower bound to the log-evidence leads to the following variational optimization problem [24]:

$$\boldsymbol{\theta}_\star \in \arg\min_{\boldsymbol{\theta}} \ell_R(\boldsymbol{\theta}) \qquad \text{with} \qquad \ell_R(\boldsymbol{\theta}) = \mathbb{E}_{q_{\boldsymbol{\theta}}(\boldsymbol{w})}(-\log p(\boldsymbol{y} \mid \boldsymbol{w})) + \mathrm{KL}(q_{\boldsymbol{\theta}}(\boldsymbol{w}) \parallel p(\boldsymbol{w}))\,. \tag{2}$$

Equation (2) is an instance of the optimization problem in Equation (1) where the optimization is over the variational parameters $\boldsymbol{\theta}$ of the posterior approximation $q_{\boldsymbol{\theta}}(\boldsymbol{w})$. In the case of a potentially misspecified prior or likelihood, the variational formulation (2) can be generalized to arbitrary loss functions ℓ and statistical distances D to the prior [25–27], such that

$$\ell_R(\boldsymbol{\theta}) = \mathbb{E}_{q_{\boldsymbol{\theta}}(\boldsymbol{w})}(\ell(\boldsymbol{y}, f_{\boldsymbol{w}}(\boldsymbol{X}))) + \lambda\, \mathrm{D}(q_{\boldsymbol{\theta}}, p). \tag{3}$$

3 Variational Deep Learning via Implicit Regularization

Our goal is to learn a variational distribution $q_{\boldsymbol{\theta}}(\boldsymbol{w})$ for the parameters $\boldsymbol{w}$ of a neural network, as in a Bayesian neural network. However, in contrast to training with (generalized) variational inference, which has an *explicit* regularization term defined via the prior to avoid overfitting, we will demonstrate how to perform variational inference over the weights of a deep neural network purely via *implicit* regularization, removing the need to store and compute quantities involving the prior entirely. We will see that this approach inherits the well-established optimization toolkit from deep learning seamlessly, while providing uncertainty quantification at minimal overhead.

3.1 Training via the Expected Loss

Rather than performing variational inference by explicitly regularizing the variational distribution to remain close to the prior, we propose to train by *minimizing the expected loss* $\bar{\ell}(\boldsymbol{\theta})$ in analogy to how deep neural networks are usually trained. Therefore, the optimal variational parameters are given by

$$\boldsymbol{\theta}_\star \in \arg\min_{\boldsymbol{\theta}} \underbrace{\mathbb{E}_{q_{\boldsymbol{\theta}}(\boldsymbol{w})}(\ell(\boldsymbol{y}, f_{\boldsymbol{w}}(\boldsymbol{X})))}_{:=\bar{\ell}(\boldsymbol{\theta})} \cancel{+ \mathrm{D}(q_{\boldsymbol{\theta}}, p)}. \tag{4}$$

Practically, this means we do not need to compute the regularization term and its gradient during training and we do not need to allocate additional memory for the prior hyperparameters.[1] However, at first glance modifying the variational objective in Eq. (2) by removing the divergence term seems to defeat the purpose of a Bayesian approach entirely. We are completely omitting the prior from the training loss. Why should the uncertainty over the weights not collapse? How does this incorporate any prior information?

3.2 Implicit Bias of (S)GD as Generalized Variational Inference

Unexpectedly, training via the expected loss achieves regularization to the prior *solely by initializing (stochastic) gradient descent to the prior*, as we will prove in Section 4 for an overparametrized linear model. Moreover, we can characterize this implicit regularization exactly. (S)GD converges to a global minimum $\boldsymbol{\theta}_\star^{\mathrm{GD}} \in \arg\min \bar{\ell}(\boldsymbol{\theta})$ of the training loss, given an appropriate learning rate sequence. But among the global minima, if (S)GD is initialized to the parameters of the prior and the Gaussian variational family is parametrized appropriately, then the solution identified by (S)GD minimizes the 2-Wasserstein distance to the prior, i.e.

$$q_{\boldsymbol{\theta}_\star^{\mathrm{GD}}} = \underset{\text{s.t. } \boldsymbol{\theta} \in \arg\min \bar{\ell}(\boldsymbol{\theta})}{\arg\min_{q_{\boldsymbol{\theta}}}} \mathrm{W}_2^2(q_{\boldsymbol{\theta}}, p)\,.$$

This equation shows that the implicit bias of (S)GD is such that it converges to a generalized variational posterior, minimizing Eq. (3) for a certain regularization strength, but with a regularizer that is *not* a KL divergence as it would be for standard variational inference, but rather a 2-Wasserstein distance to the prior. Given this characterization, we call our method Implicit Bias VI.

Section 4 provides a detailed version of the regression results introduced here and proves a similar result for binary classification. Our experiments in Section 5 focus on the application to deep neural networks. Since training via the expected loss can achieve zero loss in overparameterized models, we expect our approach to mimic vanilla deep networks closely in-distribution, while falling back to the prior out-of-distribution, as enforced by the 2-Wasserstein regularizer.

[1] We only need them to initialize the optimizer after which we can free up the memory.

3.3 Computational Efficiency

In practice, we minibatch the expected loss both over training data and parameter samples $\boldsymbol{w}_m$ drawn from the variational distribution $q_{\boldsymbol{\theta}}(\boldsymbol{w})$ such that

$$\bar{\ell}(\boldsymbol{\theta}) = \mathbb{E}_{q_{\boldsymbol{\theta}}(\boldsymbol{w})}(\ell(\boldsymbol{y}, f_{\boldsymbol{w}}(\boldsymbol{X}))) \approx \frac{1}{N_b M} \sum_{n=1}^{N_b} \sum_{m=1}^{M} \ell(\boldsymbol{y}_n, f_{\boldsymbol{w}_m}(\boldsymbol{X}_n)). \tag{5}$$

The training cost is primarily determined by two factors. The number of parameter samples M we draw for each evaluation of the objective, and the variational family, which determines the number of additional parameters of the model and the cost for sampling a set of parameters in each forward pass. We wish to keep the overhead compared to a vanilla deep neural network as small as possible.

Training With A Single Parameter Sample ($M = 1$) When drawing fewer parameter samples $\boldsymbol{w}_m$ the training objective in Eq. (5) becomes noisier in the same way a smaller batch size impacts the loss. This is concerning since the optimization procedure may not converge given this additional noise. However, one can *train with a single parameter sample only*, simply by reducing the learning rate appropriately, as we show experimentally in Figure 2 and Section S3.2. Therefore given a set of sampled parameters, the cost of a forward and backward pass is identical to a standard neural network (up to the overhead of the covariance parameters). In analogy to the previously observed relationship [e.g., 28–30] between the optimal batch size N_b, learning rate η and momentum γ, when optimizing the expected loss we conjecture the following scaling law for the optimal batch size N_b and number of parameter samples M:

$$N_b M \propto \tfrac{\eta}{1-\gamma}. \tag{6}$$

Figure 2 illustrates this. When using fewer parameter samples in the expected loss, training is unstable unless the learning rate is chosen sufficiently small. For a fixed number of optimizer steps this decreases performance, but either training for more steps, or using momentum closes this gap. As predicted by Eq. (6), momentum requires a smaller learning rate than vanilla SGD.

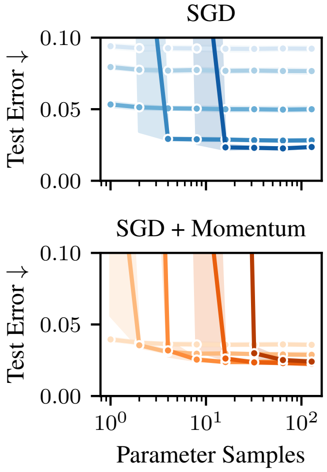


Figure 2: *Training with a single parameter sample given a small enough learning rate.* Lighter color shades correspond to smaller learning rates. See also Section S3.2.

Variational Family and Covariance Structure We choose a Gaussian variational distribution $q_{\boldsymbol{\theta}}(\boldsymbol{w})$ over (a subset of the) weights of the neural network. While at first glance this may seem restrictive, there is ample evidence that variational families in deep neural networks do not need to be complex to be expressive [31, 32]. In fact, in analogy to deep feedforward NNs with ReLU activations being universal approximators [33], one can show that Bayesian neural networks with ReLU activations and at least one Gaussian hidden layer are universal conditional distribution approximators, meaning they can approximate any continuous conditional distribution arbitrarily well [32]. As we show in Section 4, training an overparametrized linear model with SGD via the expected loss amounts to generalized variational inference *if the covariance is factorized*, i.e. $\boldsymbol{\Sigma} = \boldsymbol{S}\boldsymbol{S}^\top$ where $\boldsymbol{S} \in \mathbb{R}^{P \times R}$ is a *dense* matrix with rank $R \leq P$. Note that this (low-rank) parametrization of a covariance is non-standard in the sense that the implicit regularization result does *not* hold in the same form for a Cholesky factorization! Motivated by the theoretical observation of the implicit bias in Theorem 1, we use Gaussian layers with factorized covariances for all architectures.

3.4 Related Work

Variational inference in the context of Bayesian deep learning has seen rapid development in recent years [34–39]. Using a Wasserstein regularizer [27] in the context of generalized VI [26] is arguably most related to our work, given our theoretical results. Structure in the variational parameters has always played an important role for computational reasons [31, 40, 41] and often only a few layers are treated probabilistically [32], with some methods only considering the last layer, effectively treating the neural network as a feature extractor [42, 43]. The Laplace approximation if applied in the last-layer also falls under this category, which has the advantage that it can be applied post-hoc [13, 44–51]. Deep ensembles repeat the standard training process using multiple random initializations [52, 53] and have been linked to Bayesian methods [54, 55] with certain caveats [56, 57]. While we use SGD only to optimize the variational parameters and arguably average over samples by using momentum, SGD has also been used widely to directly approximate samples from a target distribution [54, 58–60]. Our theoretical analysis extends recent developments on the implicit bias of overparameterized linear models [4, 5, 7] to the probabilistic setting. For classification, works have focused on convergence rates [6], SGD [7], SGD with momentum [8], and the multiclass setting [10]. Results on the implicit bias of neural network training [22] often assume large widths [9, 61–64] allowing similar arguments as for linear models. The former is exemplified by the neural tangent parametrization, under which neural networks behave like kernel methods in the infinite width limit [65]. Yang et al. [63, 64, 66, 67] developed an alternative parameterization that still admits feature learning in the infinite width limit, which we extended to the case of variational networks.

4 Theoretical Analysis

Consider an overparameterized linear model with a Gaussian prior, which is trained via maximum expected log-likelihood using (stochastic) gradient descent. We will show that, in both regression (Theorem 1) and binary classification (Theorem 2), our approach can be understood as generalized variational inference with a 2-Wasserstein regularizer, which penalizes deviation from the prior. These theoretical results directly recover analogous results for non-probabilistic models [4, 5].

4.1 Linear Regression

Theorem 1 (Implicit Bias in Regression)
Let $f_{\boldsymbol{w}}(\boldsymbol{x}) = \boldsymbol{x}^\top \boldsymbol{w}$ be an overparametrized linear model with $P > N$. Define a Gaussian prior $p(\boldsymbol{w}) = \mathcal{N}\big(\boldsymbol{w}; \boldsymbol{\mu}_0, \boldsymbol{S}_0 \boldsymbol{S}_0^\top\big)$ and likelihood $p(\boldsymbol{y} \mid \boldsymbol{w}) = \mathcal{N}\big(\boldsymbol{y}; f_{\boldsymbol{w}}(\boldsymbol{X}), \sigma^2 \boldsymbol{I}\big)$ and assume a variational family $q_{\boldsymbol{\theta}}(\boldsymbol{w}) = \mathcal{N}\big(\boldsymbol{w}; \boldsymbol{\mu}, \boldsymbol{S}\boldsymbol{S}^\top\big)$ with $\boldsymbol{\theta} = (\boldsymbol{\mu}, \boldsymbol{S})$ such that $\boldsymbol{\mu} \in \mathbb{R}^P$ and $\boldsymbol{S} \in \mathbb{R}^{P \times R}$ where $R \leq P$. If the learning rate sequence $(\eta_t)_t$ is chosen such that the limit point $\boldsymbol{\theta}_\star^{\mathrm{GD}} = \lim_{t \to \infty} \boldsymbol{\theta}_t^{\mathrm{GD}}$ identified by gradient descent, initialized at $\boldsymbol{\theta}_0 = (\boldsymbol{\mu}_0, \boldsymbol{S}_0)$, is a (global) minimizer of the expected log-likelihood $\bar{\ell}(\boldsymbol{\theta})$, then

$$\boldsymbol{\theta}_\star^{\mathrm{GD}} \in \underset{\substack{\boldsymbol{\theta} = (\boldsymbol{\mu}, \boldsymbol{S}) \\ \textit{s.t. } \boldsymbol{\theta} \in \arg\min \bar{\ell}(\boldsymbol{\theta})}}{\arg\min} \quad \mathrm{W}_2^2(q_{\boldsymbol{\theta}}, p)\,. \tag{7}$$

Further, this also holds in the case of stochastic gradient descent and when using momentum.

Proof. See Section S1.1.1. □

Theorem 1 states that, among those variational parameters which minimize the expected loss, SGD (with momentum) converges to the unique variational distribution which is closest in 2-Wasserstein distance to the prior. This characterization of the implicit regularization of SGD as generalized variational inference differs from a standard ELBO objective (2) in VI via the choice of regularizer. Since the variational parameters minimize the expected loss in Equation (7), all samples from the predictive distribution interpolate the training data (see Figure 1(b), right panel), the same way a standard neural network would. In contrast, when training with a KL regularizer, the uncertainty does not collapse at the training data (see Figure 1(b), left panel), in fact a KL regularizer would diverge to infinity for a Gaussian with vanishing variance. Now, for test points that are increasingly out-of-distribution, i.e. less aligned with the span of the training data, the variational predictive matches the prior predictive more closely. Next, we will prove a similar result for binary classification.

4.2 Binary Classification of Linearly Separable Data

Consider a binary classification problem with labels $y_n \in \{-1, 1\}$, a linear model $f_{\boldsymbol{w}}(\boldsymbol{x}) = \boldsymbol{x}^\top \boldsymbol{w}$ and a variational distribution $q_{\boldsymbol{\theta}}(\boldsymbol{w})$ with variational parameters $\boldsymbol{\theta}$. The expected empirical loss is $\bar{\ell}(\boldsymbol{\theta}) = \sum_{n=1}^N \mathbb{E}_{q_{\boldsymbol{\theta}}(\boldsymbol{w})}\big(\ell(y_n \boldsymbol{x}_n^\top \boldsymbol{w})\big)$. We assume without loss of generality that all labels are positive,[2] such that $y_n = 1$, and that the dataset is linearly separable.

[2] This is not a restriction since we can always absorb the sign into the inputs, such that $\boldsymbol{x}_n' := y_n \boldsymbol{x}_n$.

Assumption 1 The dataset is *linearly separable*: $\exists \boldsymbol{w} \in \mathbb{R}^P$ such that $\forall n : \boldsymbol{w}^\top \boldsymbol{x}_n > 0$.

Define $\hat{\boldsymbol{\mu}}$ to be the L_2 *max margin vector*, i.e. the solution to the hard margin SVM:

$$\hat{\boldsymbol{\mu}} = \operatorname*{arg\,min}_{\boldsymbol{\mu} \in \mathbb{R}^P} \|\boldsymbol{\mu}\|_2^2 \quad \text{s.t.} \quad \boldsymbol{\mu}^\top \boldsymbol{x}_n \geq 1, \tag{8}$$

and the set of *support vectors* $\mathcal{S} = \arg\min_{n \in [N]} \boldsymbol{x}_n^\top \hat{\boldsymbol{\mu}}$ indexing those data points that lie on the margin. We adapt the following additional assumption from Nacson, Srebro, and Soudry [7], which can be omitted at expense of simplicity as we show in Section S1.2.

Assumption 2 The SVM support vectors span the dataset: $\operatorname{span}(\{\boldsymbol{x}_n\}_{n \in [N]}) = \operatorname{span}(\{\boldsymbol{x}_n\}_{n \in \mathcal{S}})$.

We can now characterize the implicit bias in the case of binary classification.

Theorem 2 (Implicit Bias in Binary Classification)
Let $f_{\boldsymbol{w}}(\boldsymbol{x}) = \boldsymbol{x}^\top \boldsymbol{w}$ be an (overparametrized) linear model and define a Gaussian prior $p(\boldsymbol{w}) = \mathcal{N}(\boldsymbol{w}; \boldsymbol{\mu}_0, \boldsymbol{S}_0 \boldsymbol{S}_0^\top)$. Assume a variational distribution $q_{\boldsymbol{\theta}}(\boldsymbol{w}) = \mathcal{N}(\boldsymbol{w}; \boldsymbol{\mu}, \boldsymbol{S}\boldsymbol{S}^\top)$ over the weights $\boldsymbol{w} \in \mathbb{R}^P$ with variational parameters $\boldsymbol{\theta} = (\boldsymbol{\mu}, \boldsymbol{S})$ such that $\boldsymbol{S} \in \mathbb{R}^{P \times R}$ and $R \leq P$. Assume we are using the exponential loss $\ell(u) = \exp(-u)$ and optimize the expected empirical loss $\bar{\ell}(\boldsymbol{\theta})$ via gradient descent initialized at the prior, i.e. $\boldsymbol{\theta}_0 = (\boldsymbol{\mu}_0, \boldsymbol{S}_0)$, with a sufficiently small learning rate η. Then for almost any dataset which is linearly separable (Assumption 1) and for which the support vectors span the data (Assumption 2), the rescaled gradient descent iterates (rGD)

$$\boldsymbol{\theta}_t^{\text{rGD}} = (\boldsymbol{\mu}_t^{\text{rGD}}, \boldsymbol{S}_t^{\text{rGD}}) = \left(\tfrac{1}{\log(t)} \boldsymbol{\mu}_t^{\text{GD}} + \boldsymbol{P}_{\text{null}(\boldsymbol{X})} \boldsymbol{\mu}_0, \boldsymbol{S}_t^{\text{GD}}\right) \tag{9}$$

converge to a limit point $\boldsymbol{\theta}_\star^{\text{rGD}} = \lim_{t \to \infty} \boldsymbol{\theta}_t^{\text{rGD}}$ for which it holds that

$$\boldsymbol{\theta}_\star^{\text{rGD}} \in \operatorname*{arg\,min}_{\substack{\boldsymbol{\theta} = (\boldsymbol{\mu}, \boldsymbol{S}) \\ \text{s.t. } \boldsymbol{\theta} \in \Theta_\star}} \mathrm{W}_2^2(q_{\boldsymbol{\theta}}, p)\,. \tag{10}$$

where the feasible set $\Theta_\star = \{(\boldsymbol{\mu}, \boldsymbol{S}) \mid \boldsymbol{P}_{\text{range}(\boldsymbol{X}^\top)} \boldsymbol{\mu} = \hat{\boldsymbol{\mu}} \quad \text{and} \quad \forall n : \operatorname{Var}_{q_{\boldsymbol{\theta}}}(f_{\boldsymbol{w}}(\boldsymbol{x}_n)) = 0\}$ consists of mean parameters which, if projected onto the training data, are equivalent to the L_2 max margin vector and covariance parameters such that there is no uncertainty at training data.

Proof. See Section S1.2. □

Theorem 2 states that the mean parameters $\boldsymbol{\mu}_t$ converge to the L_2 max-margin vector $\hat{\boldsymbol{\mu}}$ in the span of the training data, i.e. the data manifold, and there uncertainty collapses to zero. This is analogous to the regression case, where zero training loss enforces interpolation of the training data. In the null space of the training data, i.e. off of the data manifold, the model falls back on the prior as enforced by the 2-Wasserstein distance. The assumption of an exponential loss is standard in the literature and we expect this to extend to (binary) cross-entropy in the same way it does in results for standard neural networks [4, 6–8, 10]. Similarly, we conjecture that Theorem 2 can be extended to SGD with momentum [cf. 7, 8]. While Theorem 2 is similar to Theorem 1, there are some subtle differences. First, the feasible set for the minimization problem in Equation (10) is not the set of minima of the expected loss. This is because the exponential function does not have an optimum in contrast to a quadratic function. However, the sequence of variational parameters identified by gradient descent still satisfies $\lim_{t \to \infty} \bar{\ell}(\boldsymbol{\theta}_t) = 0$. Second, without transformation of the mean parameters, the exponential loss results in the mean parameters being unbounded. This necessitates the transformation in Equation (9) as we explain in detail in Section S1.3.

5 Experiments

We benchmark the *generalization* and *robustness* of our approach, Implicit Bias VI (IBVI), against standard neural networks and several baselines for uncertainty quantification, namely Temperature Scaling (TS) [68], Laplace approximation (LA-GS) & (LA-ML) [44, 45, 49], Weight-Space VI (WSVI) [34, 35], SWA-Gaussian (SWAG) [69] and Deep Ensembles (DE) [52], on a set of standard benchmark datasets for image classification and robustness to input corruptions. We use a convolutional architecture (either LeNet5 [70] or ResNet34 [71]) throughout, which, for all datasets but MNIST, is initialized with pretrained weights in all layers except for the input and output layer.

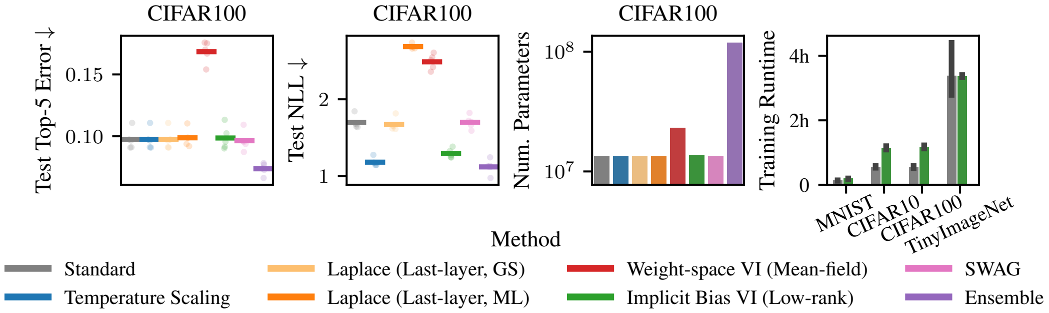


Figure 3: *In-distribution generalization and uncertainty quantification.* Implicit Bias VI (IBVI) has similar test error to other Bayesian deep learning approaches and achieves competitive uncertainty quantification on in-distribution data. While ensembles have improved accuracy, they come at an additional memory overhead. Training a probabilistic model via IBVI has only a minor computational overhead during training, both in time and memory, over standard deep learning.

All models were trained with SGD with momentum $\gamma = 0.9$ and a batch size of $N_b = 128$ for 200 epochs in single precision on an NVIDIA GH200 GPU. Results shown are averaged across five random seeds. A detailed description of the datasets, metrics, models and training can be found in Section S3. An implementation of our method is contained in the supplementary material and will be open-sourced upon publication.

In-Distribution Generalization and Uncertainty Quantification In order to assess the in-distribution generalization, we measure the test error, negative log-likelihood (NLL) and calibration error (ECE) on MNIST, CIFAR10, CIFAR100 and TinyImageNet. As Figure 3 shows for CIFAR100, and Figure S11 for all datasets, the test error for post-hoc methods (TS, LA-GS, LA-ML) is unchanged. As expected, SWAG and IBVI perform similarly with only Ensembles providing an increase in accuracy, but at substantial memory overhead compared to most other approaches. In-distribution uncertainty quantification measured in terms of NLL is improved substantially by TS, DE and IBVI with only LA and WSVI showing occasional worsening of NLL compared to the base model. The full results in Figure S11 show that TS, DE and IBVI consistently are also the best calibrated. As described in Section 3.3, for IBVI we train with a single sample only and a probabilistic input and output layer with low-rank covariance, reducing the computational overhead compared to a standard neural network to as little as $\approx 10\%$ both in time and memory (see Figure 3). See Section S3.3.2 for the full experimental results including different parametrizations (SP vs μP).

Robustness to Input Corruptions We evaluate the robustness of the different models on MNISTC [72], CIFAR10C, CIFAR100C and TinyImageNetC [73]. These are corrupted versions of the original datasets, where the images are modified via a set of 15 corruptions, such as impulse noise, blur, pixelation etc. We selected the maximum severity for each corruption and averaged the performance across all. As expected, the performance of all models drops compared to the in-distribution performance measured on the standard test sets as Figure 4 shows. Besides DE which consistently show lower test error, also IBVI shows improved accuracy on corrupted data compared to all other approaches. When using the maximal update parametrization, SWAG shows good accuracy on the two larger datasets (see Figure S13). TS, DE and IBVI perform consistently well in terms of uncertainty quantification (both for NLL and ECE) across all datasets, with LA-ML being somewhat competitive in terms of NLL. However, compared to the in-distribution setting IBVI has better uncertainty quantification than the Ensembles across all datasets.

Limitations Compared to standard neural networks, when training via Implicit Bias VI, we observed that often lower learning rates were necessary due to the additional stochasticity in the objective (see also Section 3.3). While this does not have a significant impact on generalization, the models sometimes require slightly more epochs to achieve similar in-distribution performance to standard neural networks. Effectively, in the beginning of training it takes a bit more time for IBVI to become sufficiently certain about those features which are critical for in-distribution performance. This also means that folk knowledge on learning rate settings for specific architectures may not immediately transfer. In the experiments we train models with probabilistic in- and output layers with

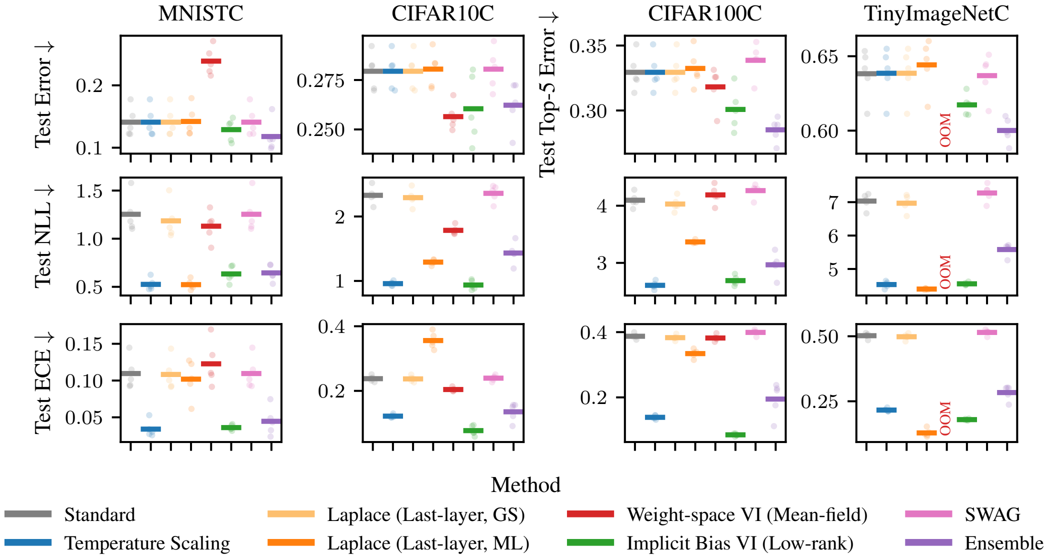


Figure 4: *Generalization on robustness benchmark problems.* When comparing different methods for Bayesian deep learning with regards to robustness to 15 different input corruptions, our approach, Implicit Bias VI, consistently has competitive uncertainty quantification across different datasets and metrics without sacrificing accuracy compared to a non-probabilistic network.

our approach, but we have so far not explored other covariance structures or where in the network probabilistic layers are most beneficial. While there is some theoretical evidence this may be sufficient [32], we believe there is potential for improvement. Beyond the prior induced by a choice of parametrization, we did not experiment with more informative or learned priors, which could potentially give significant performance improvements on certain tasks [15].

6 Conclusion

In this paper, we demonstrated how to exploit the implicit regularization of (stochastic) gradient descent for variational deep learning, as opposed to relying on explicit regularization. We rigorously characterized this implicit bias for an overparametrized linear model and showed that our approach is equivalent to generalized variational inference with a 2-Wasserstein regularizer at reduced computational cost. thus conferring desirable properties such as learning rate transfer. Lastly, we empirically demonstrated competitive performance with state-of-the-art methods for Bayesian deep learning on a set of in- and out-of-distribution benchmarks with minimal computational overhead over standard deep learning. In principle, our approach is not restricted to Gaussian variational families and should seemlessly extend to location-scale families, which could further improve performance. Finally, it would be interesting to explore connections between Implicit Bias VI and Bayesian deep learning in function-space [e.g., 27, 51, 74–76].

References

[1] M. Goldblum, M. Finzi, K. Rowan, and A. G. Wilson. “The No Free Lunch Theorem, Kolmogorov Complexity, and the Role of Inductive Biases in Machine Learning”. In: *International Conference on Machine Learning (ICML)*. 2024. DOI: 10.48550/arXiv.2304.05366 (cit. on p. 1).

[2] M. S. Nacson, R. Mulayoff, G. Ongie, T. Michaeli, and D. Soudry. “The Implicit Bias of Minima Stability in Multivariate Shallow ReLU Networks”. In: *International Conference on Learning Representations (ICLR)*. 2023. DOI: 10.48550/arXiv.2306.17499 (cit. on p. 1).

[3] R. Mulayoff, T. Michaeli, and D. Soudry. “The Implicit Bias of Minima Stability: A View from Function Space”. In: *Advances in Neural Information Processing Systems (NeurIPS)*. 2021. URL: https://proceedings.neurips.cc/paper/2021/hash/944a5ae3483ed5c1e10bbccb7942a279-Abstract.html (cit. on p. 1).

[4] D. Soudry, E. Hoffer, M. S. Nacson, S. Gunasekar, and N. Srebro. “The Implicit Bias of Gradient Descent on Separable Data”. In: *Journal of Machine Learning Research (JMLR)* (2018). DOI: 10.48550/arXiv.1710.10345 (cit. on pp. 1, 2, 5, 6, 19, 20, 23, 28).

[5] S. Gunasekar, J. Lee, D. Soudry, and N. Srebro. “Characterizing Implicit Bias in Terms of Optimization Geometry”. In: *International Conference on Machine Learning (ICML)*. 2018. DOI: 10.48550/arXiv.1802.08246 (cit. on pp. 1, 2, 5, 18).

[6] M. S. Nacson, J. D. Lee, S. Gunasekar, P. H. P. Savarese, N. Srebro, and D. Soudry. “Convergence of Gradient Descent on Separable Data”. In: *International Conference on Artificial Intelligence and Statistics (AISTATS)*. 2019. DOI: 10.48550/arXiv.1803.01905 (cit. on pp. 1, 5, 6).

[7] M. S. Nacson, N. Srebro, and D. Soudry. “Stochastic Gradient Descent on Separable Data: Exact Convergence with a Fixed Learning Rate”. In: *International Conference on Artificial Intelligence and Statistics (AISTATS)*. 2019. DOI: 10.48550/arXiv.1806.01796 (cit. on pp. 1, 2, 5, 6).

[8] B. Wang, Q. Meng, H. Zhang, R. Sun, W. Chen, Z.-M. Ma, and T.-Y. Liu. “Does Momentum Change the Implicit Regularization on Separable Data?” In: *Advances in Neural Information Processing Systems (NeurIPS)* (2022) (cit. on pp. 1, 5, 6).

[9] H. Jin and G. Montúfar. *Implicit Bias of Gradient Descent for Mean Squared Error Regression with Two-Layer Wide Neural Networks*. arXiv:2006.07356 [stat]. May 2023. DOI: 10.48550/arXiv.2006.07356 (cit. on pp. 1, 5).

[10] H. Ravi, C. Scott, D. Soudry, and Y. Wang. “The Implicit Bias of Gradient Descent on Separable Multiclass Data”. In: *Advances in Neural Information Processing Systems (NeurIPS)*. 2024. DOI: 10.48550/arXiv.2411.01350 (cit. on pp. 1, 5, 6).

[11] T. Papamarkou, M. Skoularidou, K. Palla, L. Aitchison, J. Arbel, D. Dunson, M. Filippone, V. Fortuin, P. Hennig, J. M. Hernández-Lobato, A. Hubin, A. Immer, T. Karaletsos, M. E. Khan, A. Kristiadi, Y. Li, S. Mandt, C. Nemeth, M. A. Osborne, T. G. J. Rudner, D. Rügamer, Y. W. Teh, M. Welling, A. G. Wilson, and R. Zhang. “Position: Bayesian Deep Learning is Needed in the Age of Large-Scale AI”. In: *International Conference on Machine Learning (ICML)*. 2024. DOI: 10.48550/arXiv.2402.00809 (cit. on p. 1).

[12] D. Tran, J. Liu, M. W. Dusenberry, D. Phan, M. Collier, J. Ren, K. Han, Z. Wang, Z. Mariet, H. Hu, N. Band, T. G. J. Rudner, K. Singhal, Z. Nado, J. v. Amersfoort, A. Kirsch, R. Jenatton, N. Thain, H. Yuan, K. Buchanan, K. Murphy, D. Sculley, Y. Gal, Z. Ghahramani, J. Snoek, and B. Lakshminarayanan. *Plex: Towards Reliability using Pretrained Large Model Extensions*. July 15, 2022. DOI: 10.48550/arXiv.2207.07411. arXiv: 2207.07411[cs]. URL: http://arxiv.org/abs/2207.07411 (visited on 05/16/2025) (cit. on p. 1).

[13] H. Ritter, A. Botev, and D. Barber. “Online Structured Laplace Approximations For Overcoming Catastrophic Forgetting”. In: *Advances in Neural Information Processing Systems (NeurIPS)*. 2018. DOI: 10.48550/arXiv.1805.07810 (cit. on pp. 1, 5).

[14] Y. L. Li, T. G. J. Rudner, and A. G. Wilson. “A Study of Bayesian Neural Network Surrogates for Bayesian Optimization”. In: *International Conference on Learning Representations (ICLR)*. 2024. DOI: 10.48550/arXiv.2305.20028 (cit. on p. 1).

[15] V. Fortuin. “Priors in Bayesian Deep Learning: A Review”. In: *International Statistical Review* 90.3 (2022), pp. 563–591. DOI: 10.1111/insr.12502 (cit. on pp. 1, 8).

[16] P. Izmailov, S. Vikram, M. D. Hoffman, and A. G. Wilson. "What Are Bayesian Neural Network Posteriors Really Like?" In: *International Conference on Machine Learning (ICML)*. 2021. DOI: 10.48550/arXiv.2104.14421 (cit. on p. 1).

[17] B. Adlam, J. Snoek, and S. L. Smith. *Cold Posteriors and Aleatoric Uncertainty*. July 31, 2020. DOI: 10.48550/arXiv.2008.00029. arXiv: 2008.00029[stat]. URL: http://arxiv.org/abs/2008.00029 (visited on 05/15/2025) (cit. on p. 1).

[18] T. Cinquin, A. Immer, M. Horn, and V. Fortuin. "Pathologies in priors and inference for Bayesian transformers". In: *NeurIPS Bayesian Deep Learning Workshop*. 2021. DOI: 10.48550/arXiv.2110.04020 (cit. on p. 1).

[19] B. Coker, W. P. Bruinsma, D. R. Burt, W. Pan, and F. Doshi-Velez. "Wide Mean-Field Bayesian Neural Networks Ignore the Data". In: *International Conference on Artificial Intelligence and Statistics (AISTATS)*. 2022. DOI: 10.48550/arXiv.2202.11670 (cit. on p. 1).

[20] A. Y. K. Foong, D. R. Burt, Y. Li, and R. E. Turner. "On the Expressiveness of Approximate Inference in Bayesian Neural Networks". In: *Advances in Neural Information Processing Systems (NeurIPS)*. 2020. DOI: 10.48550/arXiv.1909.00719 (cit. on p. 1).

[21] C. Zhang, S. Bengio, M. Hardt, B. Recht, and O. Vinyals. "Understanding deep learning requires rethinking generalization". In: *International Conference on Learning Representations (ICLR)*. 2017. DOI: 10.48550/arXiv.1611.03530 (cit. on p. 2).

[22] G. Vardi. "On the Implicit Bias in Deep-Learning Algorithms". In: *Commun. ACM* 66.6 (May 2023), pp. 86–93. DOI: 10.1145/3571070 (cit. on pp. 2, 5).

[23] B. Vasudeva, P. Deora, and C. Thrampoulidis. *Implicit Bias and Fast Convergence Rates for Self-attention*. 2024. DOI: 10.48550/arXiv.2402.05738 (cit. on p. 2).

[24] A. Zellner. "Optimal Information Processing and Bayes's Theorem". In: *The American Statistician* 42.4 (1988), pp. 278–280. DOI: 10.2307/2685143 (cit. on p. 2).

[25] P. G. Bissiri, C. Holmes, and S. Walker. "A General Framework for Updating Belief Distributions". In: *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 78.5 (Nov. 2016), pp. 1103–1130. ISSN: 1369-7412, 1467-9868. DOI: 10.1111/rssb.12158 (cit. on p. 3).

[26] J. Knoblauch, J. Jewson, and T. Damoulas. "An Optimization-centric View on Bayes' Rule: Reviewing and Generalizing Variational Inference". In: *Journal of Machine Learning Research (JMLR)* 23.132 (2022), pp. 1–109. ISSN: 1533-7928. URL: http://jmlr.org/papers/v23/19-1047.html (cit. on pp. 3, 4).

[27] V. D. Wild, R. Hu, and D. Sejdinovic. "Generalized Variational Inference in Function Spaces: Gaussian Measures meet Bayesian Deep Learning". In: *Advances in Neural Information Processing Systems (NeurIPS)*. Oct. 2022. DOI: 10.48550/arXiv.2205.06342 (cit. on pp. 3, 4, 8).

[28] P. Goyal, P. Dollár, R. Girshick, P. Noordhuis, L. Wesolowski, A. Kyrola, A. Tulloch, Y. Jia, and K. He. *Accurate, Large Minibatch SGD: Training ImageNet in 1 Hour*. Tech. rep. 2018. URL: http://arxiv.org/abs/1706.02677 (cit. on pp. 4, 37).

[29] S. L. Smith and Q. V. Le. "A Bayesian Perspective on Generalization and Stochastic Gradient Descent". In: *International Conference on Learning Representations (ICLR)*. 2018. DOI: 10.48550/arXiv.1710.06451 (cit. on pp. 4, 37).

[30] S. L. Smith, P.-J. Kindermans, C. Ying, and Q. V. Le. "Don't Decay the Learning Rate, Increase the Batch Size". In: *International Conference on Learning Representations (ICLR)*. 2018. DOI: 10.48550/arXiv.1711.00489 (cit. on pp. 4, 37).

[31] S. Farquhar, L. Smith, and Y. Gal. "Liberty or Depth: Deep Bayesian Neural Nets Do Not Need Complex Weight Posterior Approximations". In: *Advances in Neural Information Processing Systems (NeurIPS)*. 2020. DOI: 10.48550/arXiv.2002.03704. URL: http://arxiv.org/abs/2002.03704 (cit. on p. 4).

[32] M. Sharma, S. Farquhar, E. Nalisnick, and T. Rainforth. "Do Bayesian Neural Networks Need To Be Fully Stochastic?" In: *International Conference on Artificial Intelligence and Statistics (AISTATS)*. 2023. DOI: 10.48550/arXiv.2211.06291 (cit. on pp. 4, 8).

[33] B. Hanin and M. Sellke. *Approximating Continuous Functions by ReLU Nets of Minimal Width*. arXiv:1710.11278 [stat]. Mar. 2018. DOI: 10.48550/arXiv.1710.11278. URL: http://arxiv.org/abs/1710.11278 (cit. on p. 4).

[34] A. Graves. “Practical Variational Inference for Neural Networks”. In: *Advances in Neural Information Processing Systems (NeurIPS)*. 2011. URL: https://papers.nips.cc/paper_files/paper/2011/hash/7eb3c8be3d411e8ebfab08eba5f49632-Abstract.html (cit. on pp. 4, 6, 38).

[35] C. Blundell, J. Cornebise, K. Kavukcuoglu, and D. Wierstra. “Weight Uncertainty in Neural Networks”. In: *International Conference on Machine Learning (ICML)*. 2015. DOI: 10.48550/arXiv.1505.05424 (cit. on pp. 4, 6, 38).

[36] G. Zhang, S. Sun, D. Duvenaud, and R. Grosse. *Noisy Natural Gradient as Variational Inference*. Feb. 26, 2018. DOI: 10.48550/arXiv.1712.02390. arXiv: 1712.02390[cs]. URL: http://arxiv.org/abs/1712.02390 (visited on 05/15/2025) (cit. on p. 4).

[37] M.-N. Tran, N. Nguyen, D. Nott, and R. Kohn. *Bayesian Deep Net GLM and GLMM*. May 25, 2018. DOI: 10.48550/arXiv.1805.10157. arXiv: 1805.10157[stat]. URL: http://arxiv.org/abs/1805.10157 (visited on 05/15/2025) (cit. on p. 4).

[38] K. Osawa, S. Swaroop, A. Jain, R. Eschenhagen, R. E. Turner, R. Yokota, and M. E. Khan. “Practical Deep Learning with Bayesian Principles”. In: *Advances in Neural Information Processing Systems (NeurIPS)*. 2019. DOI: 10.48550/arXiv.1906.02506 (cit. on p. 4).

[39] Y. Shen, N. Daheim, B. Cong, P. Nickl, G. M. Marconi, C. Bazan, R. Yokota, I. Gurevych, D. Cremers, M. E. Khan, and T. Möllenhoff. “Variational Learning is Effective for Large Deep Networks”. In: *International Conference on Machine Learning (ICML)*. 2024. DOI: 10.48550/arXiv.2402.17641 (cit. on p. 4).

[40] C. Louizos and M. Welling. *Structured and Efficient Variational Deep Learning with Matrix Gaussian Posteriors*. June 23, 2016. DOI: 10.48550/arXiv.1603.04733. arXiv: 1603.04733[stat]. URL: http://arxiv.org/abs/1603.04733 (visited on 05/15/2025) (cit. on p. 4).

[41] A. Mishkin, F. Kunstner, D. Nielsen, M. Schmidt, and M. E. Khan. *SLANG: Fast Structured Covariance Approximations for Bayesian Deep Learning with Natural Gradient*. Jan. 12, 2019. DOI: 10.48550/arXiv.1811.04504. arXiv: 1811.04504[cs]. URL: http://arxiv.org/abs/1811.04504 (visited on 05/15/2025) (cit. on p. 4).

[42] J. Harrison, J. Willes, and J. Snoek. “Variational Bayesian Last Layers”. In: *International Conference on Learning Representations (ICLR)*. Apr. 2024. DOI: 10.48550/arXiv.2404.11599 (cit. on p. 4).

[43] J. Z. Liu, Z. Lin, S. Padhy, D. Tran, T. Bedrax-Weiss, and B. Lakshminarayanan. “Simple and Principled Uncertainty Estimation with Deterministic Deep Learning via Distance Awareness”. In: *Advances in Neural Information Processing Systems (NeurIPS)*. Oct. 2020. DOI: 10.48550/arXiv.2006.10108 (cit. on p. 4).

[44] D. J. C. MacKay. “A Practical Bayesian Framework for Backpropagation Networks”. In: *Neural Computation* 4 (1992). ISSN: 0899-7667, 1530-888X. DOI: 10.1162/neco.1992.4.3.448 (cit. on pp. 5, 6).

[45] H. Ritter, A. Botev, and D. Barber. “A Scalable Laplace Approximation for Neural Networks”. In: *International Conference on Learning Representations (ICLR)*. 2018 (cit. on pp. 5, 6).

[46] M. E. Khan, A. Immer, E. Abedi, and M. Korzepa. “Approximate Inference Turns Deep Networks into Gaussian Processes”. In: *Advances in Neural Information Processing Systems (NeurIPS)*. 2019. DOI: 10.48550/arXiv.1906.01930 (cit. on p. 5).

[47] A. Immer, M. Korzepa, and M. Bauer. “Improving predictions of Bayesian neural nets via local linearization”. In: *International Conference on Artificial Intelligence and Statistics (AISTATS)*. 2021 (cit. on p. 5).

[48] E. Daxberger, E. Nalisnick, J. U. Allingham, J. Antorán, and J. M. Hernández-Lobato. “Bayesian Deep Learning via Subnetwork Inference”. In: *International Conference on Machine Learning (ICML)*. 2021. DOI: 10.48550/arXiv.2010.14689 (cit. on p. 5).

[49] E. Daxberger, A. Kristiadi, A. Immer, R. Eschenhagen, M. Bauer, and P. Hennig. “Laplace Redux – Effortless Bayesian Deep Learning”. In: *Advances in Neural Information Processing Systems (NeurIPS)*. 2021. DOI: 10.48550/arXiv.2106.14806 (cit. on pp. 5, 6, 38).

[50] A. Kristiadi, A. Immer, R. Eschenhagen, and V. Fortuin. “Promises and Pitfalls of the Linearized Laplace in Bayesian Optimization”. In: *Advances in Approximate Bayesian Inference (AABI)*. 2023. DOI: 10.48550/arXiv.2304.08309 (cit. on p. 5).

[51] T. Cinquin, M. Pförtner, V. Fortuin, P. Hennig, and R. Bamler. “FSP-Laplace: Function-Space Priors for the Laplace Approximation in Bayesian Deep Learning”. In: *Advances in Neural Information Processing Systems (NeurIPS)*. Oct. 2024. DOI: 10.48550/arXiv.2407.13711. URL: http://arxiv.org/abs/2407.13711 (cit. on pp. 5, 8).

[52] B. Lakshminarayanan, A. Pritzel, and C. Blundell. “Simple and Scalable Predictive Uncertainty Estimation using Deep Ensembles”. In: *Advances in Neural Information Processing Systems (NeurIPS)*. 2017. DOI: 10.48550/arXiv.1612.01474. URL: http://arxiv.org/abs/1612.01474 (cit. on pp. 5, 6, 39).

[53] S. Fort, H. Hu, and B. Lakshminarayanan. *Deep Ensembles: A Loss Landscape Perspective*. June 25, 2020. DOI: 10.48550/arXiv.1912.02757. arXiv: 1912.02757[stat]. URL: http://arxiv.org/abs/1912.02757 (visited on 05/15/2025) (cit. on p. 5).

[54] A. G. Wilson and P. Izmailov. “Bayesian Deep Learning and a Probabilistic Perspective of Generalization”. In: *Advances in Neural Information Processing Systems (NeurIPS)*. 2020. DOI: 10.48550/arXiv.2002.08791 (cit. on p. 5).

[55] V. D. Wild, S. Ghalebikesabi, D. Sejdinovic, and J. Knoblauch. “A Rigorous Link between Deep Ensembles and (Variational) Bayesian Methods”. In: *Advances in Neural Information Processing Systems (NeurIPS)*. 2023. DOI: 10.48550/arXiv.2305.15027 (cit. on p. 5).

[56] T. Abe, E. K. Buchanan, G. Pleiss, R. Zemel, and J. P. Cunningham. “Deep Ensembles Work, But Are They Necessary?” In: *Advances in Neural Information Processing Systems (NeurIPS)*. 2022. DOI: 10.48550/arXiv.2202.06985 (cit. on p. 5).

[57] N. Dern, J. P. Cunningham, and G. Pleiss. *Theoretical Limitations of Ensembles in the Age of Overparameterization*. arXiv:2410.16201 [stat]. Oct. 2024. DOI: 10.48550/arXiv.2410.16201 (cit. on p. 5).

[58] P. Izmailov, D. Podoprikhin, T. Garipov, D. Vetrov, and A. G. Wilson. “Averaging Weights Leads to Wider Optima and Better Generalization”. In: *Conference on Uncertainty in Artificial Intelligence (UAI)*. 2018. URL: https://arxiv.org/abs/1803.05407v3 (cit. on p. 5).

[59] C. Mingard, G. Valle-Pérez, J. Skalse, and A. A. Louis. “Is SGD a Bayesian sampler? Well, almost.” In: *Journal of Machine Learning Research (JMLR)* (2020) (cit. on p. 5).

[60] J. A. Lin, J. Antorán, S. Padhy, D. Janz, J. M. Hernández-Lobato, and A. Terenin. “Sampling from Gaussian Process Posteriors using Stochastic Gradient Descent”. In: *Advances in Neural Information Processing Systems (NeurIPS)*. 2023. DOI: 10.48550/arXiv.2306.11589 (cit. on p. 5).

[61] J. Lee, L. Xiao, S. S. Schoenholz, Y. Bahri, R. Novak, J. Sohl-Dickstein, and J. Pennington. “Wide Neural Networks of Any Depth Evolve as Linear Models Under Gradient Descent”. In: *Journal of Statistical Mechanics: Theory and Experiment* 2020.12 (2020). DOI: 10.1088/1742-5468/abc62b (cit. on p. 5).

[62] J. Lai, M. Xu, R. Chen, and Q. Lin. *Generalization Ability of Wide Neural Networks on $\mathbb{R}$*. Feb. 12, 2023. DOI: 10.48550/arXiv.2302.05933. arXiv: 2302.05933[stat]. URL: http://arxiv.org/abs/2302.05933 (visited on 05/15/2025) (cit. on p. 5).

[63] G. Yang. “Tensor Programs I: Wide Feedforward or Recurrent Neural Networks of Any Architecture are Gaussian Processes”. In: *Advances in Neural Information Processing Systems (NeurIPS)*. 2019. DOI: 10.48550/arXiv.1910.12478 (cit. on p. 5).

[64] G. Yang. *Tensor Programs II: Neural Tangent Kernel for Any Architecture*. 2020. DOI: 10.48550/arXiv.2006.14548 (cit. on p. 5).

[65] A. Jacot, F. Gabriel, and C. Hongler. “Neural Tangent Kernel: Convergence and Generalization in Neural Networks”. In: *Advances in Neural Information Processing Systems (NeurIPS)*. 2018. DOI: 10.48550/arXiv.1806.07572 (cit. on p. 5).

[66] G. Yang and E. J. Hu. “Tensor Programs IV: Feature Learning in Infinite-Width Neural Networks”. In: *International Conference on Machine Learning (ICML)*. 2021. DOI: 10.48550/arXiv.2011.14522 (cit. on pp. 5, 29, 30).

[67] G. Yang, E. J. Hu, I. Babuschkin, S. Sidor, X. Liu, D. Farhi, N. Ryder, J. Pachocki, W. Chen, and J. Gao. “Tensor Programs V: Tuning Large Neural Networks via Zero-Shot Hyperparameter Transfer”. In: *Advances in Neural Information Processing Systems (NeurIPS)*. 2021. DOI: 10.48550/arXiv.2203.03466 (cit. on pp. 5, 29, 32, 34, 36).

[68] C. Guo, G. Pleiss, Y. Sun, and K. Q. Weinberger. “On Calibration of Modern Neural Networks”. In: *International Conference on Machine Learning (ICML)*. 2017. DOI: 10.48550/arXiv.1706.04599 (cit. on pp. 6, 28, 38).

[69] W. Maddox, T. Garipov, P. Izmailov, D. Vetrov, and A. G. Wilson. “A Simple Baseline for Bayesian Uncertainty in Deep Learning”. In: *Advances in Neural Information Processing Systems (NeurIPS)*. 2019. DOI: 10.48550/arXiv.1902.02476 (cit. on pp. 6, 39).

[70] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. “Gradient-based learning applied to document recognition”. In: *Proceedings of the IEEE* 86.11 (Nov. 1998), pp. 2278–2324. ISSN: 1558-2256. DOI: 10.1109/5.726791. URL: https://ieeexplore.ieee.org/document/726791 (cit. on pp. 6, 36, 37).

[71] K. He, X. Zhang, S. Ren, and J. Sun. “Deep Residual Learning for Image Recognition”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2016, pp. 770–778. ISBN: 978-1-4673-8851-1. DOI: 10.1109/CVPR.2016.90. URL: http://ieeexplore.ieee.org/document/7780459/ (cit. on pp. 6, 37).

[72] N. Mu and J. Gilmer. “MNIST-C: A Robustness Benchmark for Computer Vision”. In: *ICML Workshop on Uncertainty and Robustness in Deep Learning*. June 2019. DOI: 10.48550/arXiv.1906.02337. URL: http://arxiv.org/abs/1906.02337 (cit. on pp. 7, 36).

[73] D. Hendrycks and T. Dietterich. “Benchmarking Neural Network Robustness to Common Corruptions and Perturbations”. In: *International Conference on Learning Representations (ICLR)*. 2019. DOI: 10.48550/arXiv.1903.12261. URL: http://arxiv.org/abs/1903.12261 (cit. on pp. 7, 36).

[74] D. R. Burt, S. W. Ober, A. Garriga-Alonso, and M. van der Wilk. *Understanding Variational Inference in Function-Space*. Nov. 2020. DOI: 10.48550/arXiv.2011.09421 (cit. on p. 8).

[75] S. Qiu, T. G. J. Rudner, S. Kapoor, and A. G. Wilson. “Should We Learn Most Likely Functions or Parameters?” In: *Advances in Neural Information Processing Systems (NeurIPS)*. 2023. DOI: 10.48550/arXiv.2311.15990 (cit. on p. 8).

[76] T. G. J. Rudner, Z. Chen, Y. W. Teh, and Y. Gal. “Tractable Function-Space Variational Inference in Bayesian Neural Networks”. In: *Advances in Neural Information Processing Systems (NeurIPS)*. 2023. DOI: 10.48550/arXiv.2312.17199 (cit. on p. 8).

[77] S. P. Boyd and L. Vandenberghe. *Convex optimization*. Cambridge University Press, 2004. ISBN: 978-0-521-83378-3 (cit. on p. 14).

[78] Y. Nesterov. “A method for solving the convex programming problem with convergence rate $O(\frac{1}{k^2})$”. In: *Dokl Akad Nauk SSSR* 269 (1983), p. 543 (cit. on p. 17).

[79] B. T. Polyak. “Some methods of speeding up the convergence of iteration methods”. In: *USSR Computational Mathematics and Mathematical Physics* 4.5 (1964), pp. 1–17. DOI: 10.1016/0041-5553(64)90137-5 (cit. on p. 17).

[80] A. Bhattacharya, A. Linero, and C. J. Oates. “Grand Challenges in Bayesian Computation”. In: *Bulletin of the International Society for Bayesian Analysis (ISBA)* 31.3 (Sept. 2024). DOI: 10.48550/arXiv.2410.00496 (cit. on p. 29).

[81] A. Krizhevsky et al. *Learning multiple layers of features from tiny images*. Tech. rep. 2009 (cit. on p. 36).

[82] Y. Le and X. Yang. “Tiny ImageNet Visual Recognition Challenge”. In: *Stanford CS 231N* (2015). URL: http://cs231n.stanford.edu/tiny-imagenet-200.zip (cit. on p. 36).

[83] T. maintainers and contributors. *TorchVision: PyTorch’s Computer Vision library*. https://github.com/pytorch/vision. 2016 (cit. on p. 37).

Supplementary Material

This supplementary material contains additional results and proofs for all theoretical statements. References referring to sections, equations or theorem-type environments within this document are prefixed with ‘S’, while references to, or results from, the main paper are stated as is.

S1 Theoretical Results

Lemma S1
Let $q(\boldsymbol{w}) = \mathcal{N}(\boldsymbol{w}; \boldsymbol{\mu}, \boldsymbol{\Sigma})$, $p(\boldsymbol{w}) = \mathcal{N}(\boldsymbol{w}; \boldsymbol{\mu}_0, \boldsymbol{\Sigma}_0)$ such that $\boldsymbol{\mu}, \boldsymbol{\mu}_0 \in \mathbb{R}^P$, $\boldsymbol{\Sigma}, \boldsymbol{\Sigma}_0 \in \mathbb{R}^{P \times P}$ positive semi-definite and let $\boldsymbol{V}_A \in \mathbb{R}^{P \times N}$, $\boldsymbol{V}_B \in \mathbb{R}^{P \times (P-N)}$ be matrices with pairwise orthonormal columns that together define an orthonormal basis of $\mathbb{R}^P$, i.e. for $\boldsymbol{V} = [\boldsymbol{V}_A \quad \boldsymbol{V}_B]$ it holds that $\boldsymbol{V}\boldsymbol{V}^\top = \boldsymbol{V}^\top \boldsymbol{V} = \boldsymbol{I}$ and $\operatorname{span}(\boldsymbol{V}) = \mathbb{R}^P$. Assume further that

$$\boldsymbol{V}_A^\top \boldsymbol{\Sigma} \boldsymbol{V}_A = \boldsymbol{0}, \tag{S11}$$

then the squared 2-Wasserstein distance is given by

$$\mathrm{W}_2^2(q, p) = \left\| \boldsymbol{V}_A^\top \boldsymbol{\mu} - \boldsymbol{V}_A^\top \boldsymbol{\mu}_0 \right\|_2^2 + \mathrm{W}_2^2\big(\mathcal{N}\big(\boldsymbol{V}_B^\top \boldsymbol{\mu}, \boldsymbol{V}_B^\top \boldsymbol{\Sigma} \boldsymbol{V}_B\big), \mathcal{N}\big(\boldsymbol{V}_B^\top \boldsymbol{\mu}_0, \boldsymbol{V}_B^\top \boldsymbol{\Sigma}_0 \boldsymbol{V}_B\big)\big) + C, \tag{S12}$$

where the constant C is independent of $(\boldsymbol{\mu}, \boldsymbol{\Sigma})$.

Proof. Consider the matrix

$$\boldsymbol{V}^\top \boldsymbol{\Sigma} \boldsymbol{V} = \begin{bmatrix} \boldsymbol{0}_{N \times N} & \boldsymbol{V}_A^\top \boldsymbol{\Sigma} \boldsymbol{V}_B \\ \boldsymbol{V}_B^\top \boldsymbol{\Sigma} \boldsymbol{V}_A & \boldsymbol{V}_B^\top \boldsymbol{\Sigma} \boldsymbol{V}_B \end{bmatrix}.$$

Since $\boldsymbol{V}^\top \boldsymbol{\Sigma} \boldsymbol{V}$ is symmetric positive semi-definite, its off-diagonal block $\boldsymbol{V}_A^\top \boldsymbol{\Sigma} \boldsymbol{V}_B$ satisfies

$$(\boldsymbol{I} - \boldsymbol{0}\boldsymbol{0}^\dagger) \boldsymbol{V}_A^\top \boldsymbol{\Sigma} \boldsymbol{V}_B = \boldsymbol{0} \iff \boldsymbol{V}_A^\top \boldsymbol{\Sigma} \boldsymbol{V}_B = \boldsymbol{0}$$

by Boyd and Vandenberghe [A5.5, 77]. Therefore, we have

$$\boldsymbol{V}^\top \boldsymbol{\Sigma} \boldsymbol{V} = \begin{bmatrix} \boldsymbol{0}_{N \times N} & \boldsymbol{V}_A^\top \boldsymbol{\Sigma} \boldsymbol{V}_B \\ \boldsymbol{V}_B^\top \boldsymbol{\Sigma} \boldsymbol{V}_A & \boldsymbol{V}_B^\top \boldsymbol{\Sigma} \boldsymbol{V}_B \end{bmatrix} = \begin{bmatrix} \boldsymbol{0}_{N \times N} & \boldsymbol{0}_{N \times (P-N)} \\ \boldsymbol{0}_{(P-N) \times N} & \boldsymbol{V}_B^\top \boldsymbol{\Sigma} \boldsymbol{V}_B \end{bmatrix}. \tag{S13}$$

The squared 2-Wasserstein distance between $q(\boldsymbol{w})$ and $p(\boldsymbol{w})$ is given by

$$\mathrm{W}_2^2(q,p) = \|\boldsymbol{\mu} - \boldsymbol{\mu}_0\|_2^2 + \operatorname{tr}(\boldsymbol{\Sigma} - 2(\boldsymbol{\Sigma}^{\frac{1}{2}}\boldsymbol{\Sigma}_0\boldsymbol{\Sigma}^{\frac{1}{2}})^{\frac{1}{2}} + \boldsymbol{\Sigma}_0).$$

For the squared norm term it holds by unitary invariance of $\|\cdot\|_2$ that

$$\|\boldsymbol{\mu} - \boldsymbol{\mu}_0\|_2^2 = \|\boldsymbol{V}^\top(\boldsymbol{\mu} - \boldsymbol{\mu}_0)\|_2^2 = \left\| \begin{bmatrix} \boldsymbol{V}_A^\top(\boldsymbol{\mu} - \boldsymbol{\mu}_0) \\ \boldsymbol{V}_B^\top(\boldsymbol{\mu} - \boldsymbol{\mu}_0) \end{bmatrix} \right\|_2^2 = \left\|\boldsymbol{V}_A^\top\boldsymbol{\mu} - \boldsymbol{V}_A^\top\boldsymbol{\mu}_0\right\|_2^2 + \left\|\boldsymbol{V}_B^\top\boldsymbol{\mu} - \boldsymbol{V}_B^\top\boldsymbol{\mu}_0\right\|_2^2.$$

Now for the trace term we have that

$$\begin{aligned}
&\operatorname{tr}(\boldsymbol{V}\boldsymbol{V}^\top(\boldsymbol{\Sigma} - 2(\boldsymbol{\Sigma}^{\frac{1}{2}}\boldsymbol{\Sigma}_0\boldsymbol{\Sigma}^{\frac{1}{2}})^{\frac{1}{2}} + \boldsymbol{\Sigma}_0)) \\
&= \operatorname{tr}(\boldsymbol{V}^\top\boldsymbol{\Sigma}\boldsymbol{V}) - 2\operatorname{tr}(\boldsymbol{V}^\top(\boldsymbol{\Sigma}^{\frac{1}{2}}\boldsymbol{\Sigma}_0\boldsymbol{\Sigma}^{\frac{1}{2}})^{\frac{1}{2}}\boldsymbol{V}) + \operatorname{tr}(\boldsymbol{V}^\top\boldsymbol{\Sigma}_0\boldsymbol{V}) \\
&= \operatorname{tr}(\boldsymbol{V}_A^\top\boldsymbol{\Sigma}\boldsymbol{V}_A) + \operatorname{tr}(\boldsymbol{V}_B^\top\boldsymbol{\Sigma}\boldsymbol{V}_B) + \operatorname{tr}(\boldsymbol{V}_A^\top\boldsymbol{\Sigma}_0\boldsymbol{V}_A) + \operatorname{tr}(\boldsymbol{V}_B^\top\boldsymbol{\Sigma}_0\boldsymbol{V}_B) - 2\operatorname{tr}(\boldsymbol{V}^\top(\boldsymbol{\Sigma}^{\frac{1}{2}}\boldsymbol{\Sigma}_0\boldsymbol{\Sigma}^{\frac{1}{2}})^{\frac{1}{2}}\boldsymbol{V}) \\
&\overset{+c}{=} \operatorname{tr}(\boldsymbol{V}_B^\top\boldsymbol{\Sigma}\boldsymbol{V}_B) + \operatorname{tr}(\boldsymbol{V}_B^\top\boldsymbol{\Sigma}_0\boldsymbol{V}_B) - 2\operatorname{tr}(\boldsymbol{V}^\top(\boldsymbol{\Sigma}^{\frac{1}{2}}\boldsymbol{\Sigma}_0\boldsymbol{\Sigma}^{\frac{1}{2}})^{\frac{1}{2}}\boldsymbol{V})
\end{aligned} \tag{S14}$$

where we used Eq. (S11) and $\overset{+c}{=}$ denotes equality up to constants independent of $(\boldsymbol{\mu}, \boldsymbol{\Sigma})$.

Now by Eq. (S13), we have that $\boldsymbol{\Sigma} = \boldsymbol{V}_B\boldsymbol{M}\boldsymbol{V}_B^\top$ for $\boldsymbol{M} = \boldsymbol{V}_B^\top\boldsymbol{\Sigma}\boldsymbol{V}_B$ and its unique principal square root is given by $\boldsymbol{\Sigma}^{\frac{1}{2}} = \boldsymbol{V}_B\boldsymbol{M}^{\frac{1}{2}}\boldsymbol{V}_B^\top$ since

$$(\boldsymbol{V}_B\boldsymbol{M}^{\frac{1}{2}}\boldsymbol{V}_B^\top)(\boldsymbol{V}_B\boldsymbol{M}^{\frac{1}{2}}\boldsymbol{V}_B^\top) = \boldsymbol{V}_B\boldsymbol{M}^{\frac{1}{2}}\boldsymbol{I}_{(P-N)\times(P-N)}\boldsymbol{M}^{\frac{1}{2}}\boldsymbol{V}_B^\top = \boldsymbol{\Sigma}.$$

It also holds that the unique principal square root

$$(\boldsymbol{\Sigma}^{\frac{1}{2}}\boldsymbol{\Sigma}_0\boldsymbol{\Sigma}^{\frac{1}{2}})^{\frac{1}{2}} = \boldsymbol{V}_B(\boldsymbol{M}^{\frac{1}{2}}\boldsymbol{V}_B^\top\boldsymbol{\Sigma}_0\boldsymbol{V}_B\boldsymbol{M}^{\frac{1}{2}})^{\frac{1}{2}}\boldsymbol{V}_B^\top$$

since direct calculation gives

$$\begin{aligned}
&(\boldsymbol{V}_B(\boldsymbol{M}^{\frac{1}{2}}\boldsymbol{V}_B^\top\boldsymbol{\Sigma}_0\boldsymbol{V}_B\boldsymbol{M}^{\frac{1}{2}})^{\frac{1}{2}}\boldsymbol{V}_B^\top)(\boldsymbol{V}_B(\boldsymbol{M}^{\frac{1}{2}}\boldsymbol{V}_B^\top\boldsymbol{\Sigma}_0\boldsymbol{V}_B\boldsymbol{M}^{\frac{1}{2}})^{\frac{1}{2}}\boldsymbol{V}_B^\top) \\
&\qquad = \boldsymbol{V}_B\boldsymbol{M}^{\frac{1}{2}}\boldsymbol{V}_B^\top\boldsymbol{\Sigma}_0\boldsymbol{V}_B\boldsymbol{M}^{\frac{1}{2}}\boldsymbol{V}_B^\top = \boldsymbol{\Sigma}^{\frac{1}{2}}\boldsymbol{\Sigma}_0\boldsymbol{\Sigma}^{\frac{1}{2}}.
\end{aligned}$$

Therefore we have that

$$\operatorname{tr}(\boldsymbol{V}^\top(\boldsymbol{\Sigma}^{\frac{1}{2}}\boldsymbol{\Sigma}_0\boldsymbol{\Sigma}^{\frac{1}{2}})^{\frac{1}{2}}\boldsymbol{V}) = \operatorname{tr}(\boldsymbol{V}^\top\boldsymbol{V}_B(\boldsymbol{M}^{\frac{1}{2}}\boldsymbol{V}_B^\top\boldsymbol{\Sigma}_0\boldsymbol{V}_B\boldsymbol{M}^{\frac{1}{2}})^{\frac{1}{2}}\boldsymbol{V}_B^\top\boldsymbol{V}) = \operatorname{tr}((\boldsymbol{M}^{\frac{1}{2}}\boldsymbol{V}_B^\top\boldsymbol{\Sigma}_0\boldsymbol{V}_B\boldsymbol{M}^{\frac{1}{2}})^{\frac{1}{2}}).$$

Putting it all together we obtain

$$\begin{aligned}
\mathrm{W}_2^2(q,p) &\overset{+c}{=} \left\|\boldsymbol{V}_A^\top\boldsymbol{\mu} - \boldsymbol{V}_A^\top\boldsymbol{\mu}_0\right\|_2^2 + \left\|\boldsymbol{V}_B^\top\boldsymbol{\mu} - \boldsymbol{V}_B^\top\boldsymbol{\mu}_0\right\|_2^2 + \operatorname{tr}(\boldsymbol{V}_B^\top\boldsymbol{\Sigma}\boldsymbol{V}_B) + \operatorname{tr}(\boldsymbol{V}_B^\top\boldsymbol{\Sigma}_0\boldsymbol{V}_B) - 2\operatorname{tr}(\boldsymbol{V}^\top(\boldsymbol{\Sigma}^{\frac{1}{2}}\boldsymbol{\Sigma}_0\boldsymbol{\Sigma}^{\frac{1}{2}})^{\frac{1}{2}}\boldsymbol{V}) \\
&= \left\|\boldsymbol{V}_A^\top\boldsymbol{\mu} - \boldsymbol{V}_A^\top\boldsymbol{\mu}_0\right\|_2^2 + \left\|\boldsymbol{V}_B^\top\boldsymbol{\mu} - \boldsymbol{V}_B^\top\boldsymbol{\mu}_0\right\|_2^2 + \operatorname{tr}(\boldsymbol{V}_B^\top\boldsymbol{\Sigma}\boldsymbol{V}_B) + \operatorname{tr}(\boldsymbol{V}_B^\top\boldsymbol{\Sigma}_0\boldsymbol{V}_B) - 2\operatorname{tr}((\boldsymbol{M}^{\frac{1}{2}}\boldsymbol{V}_B^\top\boldsymbol{\Sigma}_0\boldsymbol{V}_B\boldsymbol{M}^{\frac{1}{2}})^{\frac{1}{2}}) \\
&= \left\|\boldsymbol{V}_A^\top\boldsymbol{\mu} - \boldsymbol{V}_A^\top\boldsymbol{\mu}_0\right\|_2^2 + \mathrm{W}_2^2(\mathcal{N}(\boldsymbol{V}_B^\top\boldsymbol{\mu}, \boldsymbol{V}_B^\top\boldsymbol{\Sigma}\boldsymbol{V}_B), \mathcal{N}(\boldsymbol{V}_B^\top\boldsymbol{\mu}_0, \boldsymbol{V}_B^\top\boldsymbol{\Sigma}_0\boldsymbol{V}_B))
\end{aligned}$$

which completes the proof. □

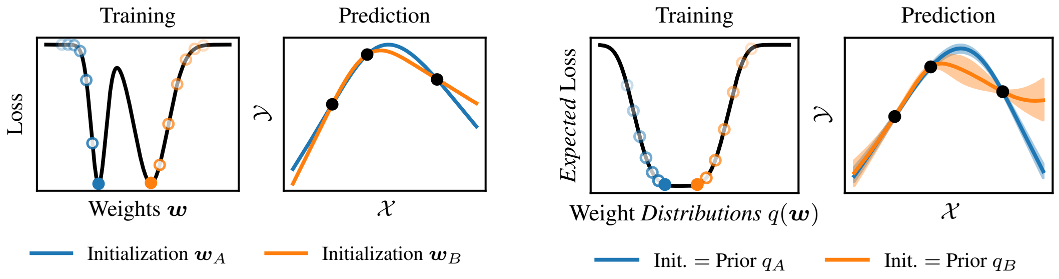
S1.1 Overparametrized Linear Regression

S1.1.1 Characterization of Implicit Bias (Proof of Theorem 1)

Theorem 1 (Implicit Bias in Regression)
Let $f_{\boldsymbol{w}}(\boldsymbol{x}) = \boldsymbol{x}^\top\boldsymbol{w}$ be an overparametrized linear model with $P > N$. Define a Gaussian prior $p(\boldsymbol{w}) = \mathcal{N}(\boldsymbol{w}; \boldsymbol{\mu}_0, \boldsymbol{S}_0\boldsymbol{S}_0^\top)$ and likelihood $p(\boldsymbol{y} \mid \boldsymbol{w}) = \mathcal{N}(\boldsymbol{y}; f_{\boldsymbol{w}}(\boldsymbol{X}), \sigma^2\boldsymbol{I})$ and assume a variational family $q_{\boldsymbol{\theta}}(\boldsymbol{w}) = \mathcal{N}(\boldsymbol{w}; \boldsymbol{\mu}, \boldsymbol{S}\boldsymbol{S}^\top)$ with $\boldsymbol{\theta} = (\boldsymbol{\mu}, \boldsymbol{S})$ such that $\boldsymbol{\mu} \in \mathbb{R}^P$ and $\boldsymbol{S} \in \mathbb{R}^{P\times R}$ where $R \leq P$. If the learning rate sequence $(\eta_t)_t$ is chosen such that the limit point $\boldsymbol{\theta}_\star^{\mathrm{GD}} = \lim_{t\to\infty}\boldsymbol{\theta}_t^{\mathrm{GD}}$ identified by gradient descent, initialized at $\boldsymbol{\theta}_0 = (\boldsymbol{\mu}_0, \boldsymbol{S}_0)$, is a (global) minimizer of the expected log-likelihood $\bar{\ell}(\boldsymbol{\theta})$, then

$$\boldsymbol{\theta}_\star^{\mathrm{GD}} \in \underset{\substack{\boldsymbol{\theta}=(\boldsymbol{\mu},\boldsymbol{S}) \\ \text{s.t. } \boldsymbol{\theta}\in\arg\min\bar{\ell}(\boldsymbol{\theta})}}{\arg\min} \mathrm{W}_2^2(q_{\boldsymbol{\theta}}, p)\,. \tag{7}$$

Further, this also holds in the case of stochastic gradient descent and when using momentum.



(a) NN trained with *no explicit regularization.*

(b) BNN trained with *no explicit regularization.*

Figure S1: *Implicit regularization in standard neural networks versus in probabilistic networks.* Left panels: A neural network trained without explicit regularization can converge to different global minima of the loss. Optimization of the weights will implicitly regularize towards one or the other. Right panels: Analogously, there are multiple distributions over neural networks that are global minima of the *expected* loss. Optimization of the *distribution* over the weights will implicitly regularize towards one or the other. Our approach uses this implicit regularization instead of an explicit regularization to a prior.

Proof. Let $\boldsymbol{\theta}_\star = (\boldsymbol{\mu}_\star, \boldsymbol{S}_\star)$ be a minimizer of $\bar{\ell}(\boldsymbol{\theta})$. By assumption it holds that the expected negative log-likelihood is equal to the following non-negative loss function up to an additive constant:

$$\begin{aligned}\bar{\ell}(\boldsymbol{\theta}) &= \mathbb{E}_{q_{\boldsymbol{\theta}}(\boldsymbol{w})}(\ell(\boldsymbol{y}, f_{\boldsymbol{w}}(\boldsymbol{X}))) = \mathbb{E}_{q_{\boldsymbol{\theta}}(\boldsymbol{w})}(-\log p(\boldsymbol{y} \mid \boldsymbol{w}))\\ &\overset{\pm c}{=} \frac{1}{2\sigma^2}\,\mathbb{E}_{q_{\boldsymbol{\theta}}(\boldsymbol{w})}\big(\|\boldsymbol{y} - \boldsymbol{X}\boldsymbol{w}\|_2^2\big)\\ &= \frac{1}{2\sigma^2}\big(\|\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\mu}\|_2^2 + \operatorname{tr}(\boldsymbol{X}\boldsymbol{\Sigma}\boldsymbol{X}^\top)\big) \geq 0,\end{aligned}$$

where $\boldsymbol{\Sigma} = \boldsymbol{S}\boldsymbol{S}^\top$ and non-negativity follows from $\boldsymbol{\Sigma}$ being symmetric positive semi-definite. Therefore any (global) minimizer $\boldsymbol{\theta}_\star = (\boldsymbol{\mu}_\star, \boldsymbol{\Sigma}_\star)$ necessarily satisfies

$$\|\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\mu}_\star\|_2^2 = 0, \tag{S15}$$

$$\operatorname{tr}(\boldsymbol{X}\boldsymbol{\Sigma}_\star\boldsymbol{X}^\top) = 0. \tag{S16}$$

Let $\boldsymbol{V} = [\boldsymbol{V}_{\text{range}} \quad \boldsymbol{V}_{\text{null}}] \in \mathbb{R}^{P \times P}$ be the orthonormal matrix of right singular vectors of $\boldsymbol{X} = \boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{V}^\top$, where $\boldsymbol{V}_{\text{range}} \in \mathbb{R}^{P \times N}$ and $\boldsymbol{V}_{\text{null}} \in \mathbb{R}^{P \times (P-N)}$. Since $\boldsymbol{X} \in \mathbb{R}^{N \times P}$ and we are in the overparametrized regime, i.e. $P > N$, the optimal mean parameter decomposes into the least-squares solution and a null space contribution

$$\boldsymbol{\mu}_\star = \boldsymbol{V}_{\text{range}}\boldsymbol{u}_\star + \boldsymbol{V}_{\text{null}}\boldsymbol{z} = \boldsymbol{X}^\dagger\boldsymbol{y} + \boldsymbol{V}_{\text{null}}\boldsymbol{z}. \tag{S17}$$

Furthermore, it holds for positive semi-definite $\boldsymbol{\Sigma} \in \mathbb{R}^{P \times P}$ that

$$\begin{aligned}0 \leq \operatorname{tr}(\boldsymbol{X}\boldsymbol{\Sigma}\boldsymbol{X}^\top) &= \operatorname{tr}(\boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{V}^\top\boldsymbol{\Sigma}\boldsymbol{V}\boldsymbol{\Lambda}\boldsymbol{U}^\top) = \operatorname{tr}(\boldsymbol{\Lambda}\boldsymbol{V}^\top\boldsymbol{\Sigma}\boldsymbol{V}\boldsymbol{\Lambda})\\ &= \operatorname{tr}([\boldsymbol{\Lambda}_{N \times N} \quad \boldsymbol{0}] \begin{bmatrix}\boldsymbol{V}_{\text{range}}^\top\boldsymbol{\Sigma}\boldsymbol{V}_{\text{range}} & * \\ * & *\end{bmatrix} \begin{bmatrix}\boldsymbol{\Lambda}_{N \times N} \\ \boldsymbol{0}\end{bmatrix})\\ &= \operatorname{tr}(\boldsymbol{\Lambda}_{N \times N}\boldsymbol{V}_{\text{range}}^\top\boldsymbol{\Sigma}\boldsymbol{V}_{\text{range}}\boldsymbol{\Lambda}_{N \times N})\\ &= \sum_{i=1}^{N} \lambda_i^2 [\boldsymbol{V}_{\text{range}}^\top\boldsymbol{\Sigma}\boldsymbol{V}_{\text{range}}]_{ii}\end{aligned}$$

where $\lambda_i^2 > 0$ are the squared singular values of $\boldsymbol{X}$, which are strictly positive since $\operatorname{rank}(\boldsymbol{X}) = N$. Therefore using Equation (S16) any global minimizer necessarily satisfies $[\boldsymbol{V}_{\text{range}}^\top\boldsymbol{\Sigma}_\star\boldsymbol{V}_{\text{range}}]_{ii} = 0$ for $i \in \{1, \dots, N\}$. Now since $\boldsymbol{V}_{\text{range}}^\top\boldsymbol{\Sigma}_\star\boldsymbol{V}_{\text{range}}$ is symmetric positive semi-definite and its diagonal is zero, so is its trace and therefore the sum of its non-negative eigenvalues is necessarily zero. Thus all eigenvalues are zero and therefore

$$\boldsymbol{V}_{\text{range}}^\top\boldsymbol{\Sigma}\boldsymbol{V}_{\text{range}} = \boldsymbol{0}. \tag{S18}$$

Now by Lemma S1 we have that the squared 2-Wasserstein distance between $q_{\boldsymbol{\theta}_\star}(\boldsymbol{w}) = \mathcal{N}(\boldsymbol{w}; \boldsymbol{\mu}_\star, \boldsymbol{\Sigma}_\star)$ and the initialization $p(\boldsymbol{w}) = \mathcal{N}(\boldsymbol{w}; \boldsymbol{\mu}_0, \boldsymbol{\Sigma}_0)$ is given up to a constant independent of $(\boldsymbol{\mu}_\star, \boldsymbol{\Sigma}_\star)$ by

$$
\begin{aligned}
\mathrm{W}_2(q_{\boldsymbol{\theta}_\star}, p) &\overset{\pm c}{=} \left\| \boldsymbol{V}_{\text{range}}^\top \boldsymbol{\mu}_\star - \boldsymbol{V}_{\text{range}}^\top \boldsymbol{\mu}_0 \right\|_2^2 + \mathrm{W}_2^2\big(\mathcal{N}\big(\boldsymbol{V}_{\text{null}}^\top \boldsymbol{\mu}_\star, \boldsymbol{V}_{\text{null}}^\top \boldsymbol{\Sigma}_\star \boldsymbol{V}_{\text{null}}\big), \mathcal{N}\big(\boldsymbol{V}_{\text{null}}^\top \boldsymbol{\mu}_0, \boldsymbol{V}_{\text{null}}^\top \boldsymbol{\Sigma}_0 \boldsymbol{V}_{\text{null}}\big)\big) \\
&= \left\| \boldsymbol{X}^\dagger \boldsymbol{y} - \boldsymbol{V}_{\text{range}}^\top \boldsymbol{\mu}_0 \right\|_2^2 + \mathrm{W}_2^2\big(\mathcal{N}\big(\boldsymbol{V}_{\text{null}}^\top \boldsymbol{\mu}_\star, \boldsymbol{V}_{\text{null}}^\top \boldsymbol{\Sigma}_\star \boldsymbol{V}_{\text{null}}\big), \mathcal{N}\big(\boldsymbol{V}_{\text{null}}^\top \boldsymbol{\mu}_0, \boldsymbol{V}_{\text{null}}^\top \boldsymbol{\Sigma}_0 \boldsymbol{V}_{\text{null}}\big)\big) \\
&\overset{\pm c}{=} \mathrm{W}_2^2\big(\mathcal{N}\big(\boldsymbol{V}_{\text{null}}^\top \boldsymbol{\mu}_\star, \boldsymbol{V}_{\text{null}}^\top \boldsymbol{\Sigma}_\star \boldsymbol{V}_{\text{null}}\big), \mathcal{N}\big(\boldsymbol{V}_{\text{null}}^\top \boldsymbol{\mu}_0, \boldsymbol{V}_{\text{null}}^\top \boldsymbol{\Sigma}_0 \boldsymbol{V}_{\text{null}}\big)\big)
\end{aligned}
$$

Therefore among variational distributions $q_{\boldsymbol{\theta}_\star}$ with parameters $\boldsymbol{\theta}_\star$ that minimize the expected loss $\bar{\ell}(\boldsymbol{\theta})$, any such $\boldsymbol{\theta}_\star$ that minimizes the squared 2-Wasserstein distance to the prior satisfies

$$
(\underbrace{\boldsymbol{V}_{\text{null}}^\top \boldsymbol{\mu}_\star}_{=:\boldsymbol{z}}, \underbrace{\boldsymbol{V}_{\text{null}}^\top \boldsymbol{\Sigma}_\star \boldsymbol{V}_{\text{null}}}_{=:\boldsymbol{M}}) = (\boldsymbol{V}_{\text{null}}^\top \boldsymbol{\mu}_0, \boldsymbol{V}_{\text{null}}^\top \boldsymbol{\Sigma}_0 \boldsymbol{V}_{\text{null}}). \tag{S19}
$$

(Stochastic) Gradient Descent It remains to show that (stochastic) gradient descent identifies a minimum of the expected loss $\bar{\ell}(\boldsymbol{\theta})$, such that the above holds. By assumption we have for the loss on a batch $\boldsymbol{X}_b$ of data that

$$
\begin{aligned}
\bar{\ell}(\boldsymbol{\theta}) &= \mathbb{E}_{q_{\boldsymbol{\theta}}(\boldsymbol{w})}(\ell(\boldsymbol{y}_b, f_{\boldsymbol{w}}(\boldsymbol{X}_b))) = \mathbb{E}_{q_{\boldsymbol{\theta}}(\boldsymbol{w})}(-\log p(\boldsymbol{y}_b \mid \boldsymbol{w})) \\
&\overset{\pm c}{=} \frac{1}{2\sigma^2}\big(\|\boldsymbol{y}_b - \boldsymbol{X}_b \boldsymbol{\mu}\|_2^2 + \operatorname{tr}(\boldsymbol{X}_b \boldsymbol{\Sigma} \boldsymbol{X}_b^\top)\big),
\end{aligned}
$$

Therefore, at convergence of (stochastic) gradient descent the variational parameters $\boldsymbol{\theta}_\infty = (\boldsymbol{\mu}_\infty, \boldsymbol{S}_\infty)$ are given by

$$
\boldsymbol{\mu}_\infty = \boldsymbol{\mu}_0 - \sum_{t=1}^\infty \eta_t \nabla_{\boldsymbol{\mu}} \bar{\ell}_b(\boldsymbol{\theta}_{t-1}) = \boldsymbol{\mu}_0 + \sum_{t=1}^\infty \frac{\eta_t}{\sigma^2} \boldsymbol{X}_b^\top (\boldsymbol{y}_b - \boldsymbol{X}_b \boldsymbol{\mu}_{t-1})
$$

as well as

$$
\boldsymbol{S}_\infty = \boldsymbol{S}_0 - \sum_{t=1}^\infty \eta_t \nabla_{\boldsymbol{S}} \bar{\ell}_b(\boldsymbol{\theta}_{t-1}) = \boldsymbol{S}_0 - \sum_{t=1}^\infty \frac{\eta_t}{\sigma^2} \boldsymbol{X}_b^\top \boldsymbol{X}_b \boldsymbol{S}_{t-1}
$$

and therefore

$$
\begin{aligned}
\boldsymbol{z}_\infty = \boldsymbol{V}_{\text{null}}^\top \boldsymbol{\mu}_\infty &= \boldsymbol{V}_{\text{null}}^\top \boldsymbol{\mu}_0 + \sum_{t=1}^\infty \frac{\eta_t}{\sigma^2} \boldsymbol{V}_{\text{null}}^\top \underbrace{\boldsymbol{X}_b^\top (\boldsymbol{y}_b - \boldsymbol{X}_b \boldsymbol{\mu}_{t-1})}_{\in \operatorname{range}(\boldsymbol{X}_b^\top)} = \boldsymbol{V}_{\text{null}}^\top \boldsymbol{\mu}_0 \\
\boldsymbol{V}_{\text{null}}^\top \boldsymbol{S}_\infty = \boldsymbol{V}_{\text{null}}^\top \boldsymbol{S}_0 &- \sum_{t=1}^\infty \frac{\eta_t}{\sigma^2} \boldsymbol{V}_{\text{null}}^\top \underbrace{\boldsymbol{X}_b^\top \boldsymbol{X}_b \boldsymbol{S}_{t-1}}_{\text{columns } \in \operatorname{range}(\boldsymbol{X}_b^\top)} = \boldsymbol{V}_{\text{null}}^\top \boldsymbol{S}_0
\end{aligned}
$$

where we used continuity of linear maps between finite-dimensional spaces. It follows that

$$
\boldsymbol{M}_\infty = \boldsymbol{V}_{\text{null}}^\top \boldsymbol{\Sigma}_\infty \boldsymbol{V}_{\text{null}} = \boldsymbol{V}_{\text{null}}^\top \boldsymbol{S}_\infty \boldsymbol{S}_\infty^\top \boldsymbol{V}_{\text{null}} = \boldsymbol{V}_{\text{null}}^\top \boldsymbol{S}_0 \boldsymbol{S}_0^\top \boldsymbol{V}_{\text{null}} = \boldsymbol{V}_{\text{null}}^\top \boldsymbol{\Sigma}_0 \boldsymbol{V}_{\text{null}}.
$$

Therefore any limit point of (stochastic) gradient descent that minimizes the expected log-likelihood also minimizes the 2-Wasserstein distance to the prior, since $\boldsymbol{\theta}_\infty$ satisfies Equation (S19).

Momentum In case we are using (stochastic) gradient descent with momentum, the updates are given by

$$
\begin{aligned}
\boldsymbol{\mu}_{t+1} &= \boldsymbol{\mu}_t + \gamma_t \Delta \boldsymbol{\mu}_t - \eta_t \nabla_{\boldsymbol{\mu}} \bar{\ell}_b(\boldsymbol{\theta}_t + \alpha_t \Delta \boldsymbol{\theta}_t) \\
\boldsymbol{S}_{t+1} &= \boldsymbol{S}_t + \gamma_t \Delta \boldsymbol{S}_t - \eta_t \nabla_{\boldsymbol{S}} \bar{\ell}_b(\boldsymbol{\theta}_t + \alpha_t \Delta \boldsymbol{\theta}_t)
\end{aligned} \tag{S20}
$$

where

$$
\Delta \boldsymbol{\theta}_t = \begin{pmatrix} \Delta \boldsymbol{\mu}_t \\ \Delta \boldsymbol{S}_t \end{pmatrix} = \boldsymbol{\theta}_t - \boldsymbol{\theta}_{t-1}, \qquad \Delta \boldsymbol{\theta}_0 = \boldsymbol{0}.
$$

for parameters $\gamma_t, \alpha_t \geq 0$, which includes Nesterov's acceleration ($\gamma_t = \alpha_t$) [78] and heavy ball momentum ($\alpha_t = 0$) [79].

To prove that the updates of the variational parameters are always orthogonal to the null space of $\boldsymbol{X}_b$, we proceed by induction. The base case is trivial since $\Delta\boldsymbol{\theta}_0 = \mathbf{0}$. Assume now that $\boldsymbol{V}_{\text{null}}^{\top}\Delta\boldsymbol{\mu}_t = \mathbf{0}$ and $\boldsymbol{V}_{\text{null}}^{\top}\Delta\boldsymbol{S}_t = \mathbf{0}$, then by Equation (S20), we have

$$
\begin{aligned}
\boldsymbol{V}_{\text{null}}^{\top}\Delta\boldsymbol{\mu}_{t+1} &= \boldsymbol{V}_{\text{null}}^{\top}(\boldsymbol{\mu}_{t+1} - \boldsymbol{\mu}_t) = \gamma_t \boldsymbol{V}_{\text{null}}^{\top}\Delta\boldsymbol{\mu}_t - \eta_t \boldsymbol{V}_{\text{null}}^{\top}\nabla_{\boldsymbol{\mu}}\bar{\ell}_b(\boldsymbol{\theta}_t + \alpha_t\Delta\boldsymbol{\theta}_t) = \mathbf{0}\\
\boldsymbol{V}_{\text{null}}^{\top}\Delta\boldsymbol{S}_{t+1} &= \boldsymbol{V}_{\text{null}}^{\top}(\boldsymbol{S}_{t+1} - \boldsymbol{S}_t) = \gamma_t \boldsymbol{V}_{\text{null}}^{\top}\Delta\boldsymbol{S}_t - \eta_t \boldsymbol{V}_{\text{null}}^{\top}\nabla_{\boldsymbol{S}}\bar{\ell}_b(\boldsymbol{\theta}_t + \alpha_t\Delta\boldsymbol{\theta}_t) = \mathbf{0}
\end{aligned}
$$

where we used the induction hypothesis and the fact that the gradients are orthogonal to the null space as shown earlier.

Therefore by the same argument as above we have that $\boldsymbol{\theta}_\infty$ computed via (stochastic) gradient descent with momentum satisfies Equation (S19), which directly implies Theorem 1. □

S1.1.2 Connection to Ensembles

Proposition S1 (Connection to Ensembles)
Consider an ensemble of overparametrized linear models $f_{\boldsymbol{w}}(\boldsymbol{x}) = \boldsymbol{x}^{\top}\boldsymbol{w}$ initialized with weights drawn from the prior $\boldsymbol{w}_0^{(i)} \sim \mathcal{N}\big(\boldsymbol{w}; \boldsymbol{\mu}_0, \boldsymbol{S}_0\boldsymbol{S}_0^{\top}\big)$. Assume each model is trained independently to convergence via (S)GD such that $\boldsymbol{w}_\star^{(i)} = \arg\min_{\boldsymbol{w}} \ell(\boldsymbol{y}, f_{\boldsymbol{w}}(\boldsymbol{X}))$. Then the distribution over the weights of the trained ensemble $q_{\text{Ens}}(\boldsymbol{w})$ is equal to the variational approximation $q_{\boldsymbol{\theta}_\star}(\boldsymbol{w})$ learned via (S)GD initialized at the prior hyperparameters $\boldsymbol{\theta}_0 = (\boldsymbol{\mu}_0, \boldsymbol{S}_0)$, i.e.

$$
q_{\text{Ens}}(\boldsymbol{w}) = q_{\boldsymbol{\theta}_\star^{\text{GD}}}(\boldsymbol{w}). \tag{S21}
$$

Proof. The parameters $\boldsymbol{w}_\infty^{(i)}$ of the (independently) trained ensemble members identified via (stochastic) gradient descent are given by

$$
\boldsymbol{w}_\infty^{(i)} = \operatorname*{arg\,min}_{\boldsymbol{w}\in F} \|\boldsymbol{w} - \boldsymbol{w}_0^{(i)}\|_2
$$

where $F = \{\boldsymbol{w} \in \mathbb{R}^P \mid f_{\boldsymbol{w}}(\boldsymbol{X}) = \boldsymbol{X}\boldsymbol{w} = \boldsymbol{y}\}$ is the set of interpolating solutions [5, Sec. 2.1]. Since we can write F equivalently via the minimum norm solution and an arbitrary null space contribution, s.t. $F = \{\boldsymbol{w} = \boldsymbol{X}^{\dagger}\boldsymbol{y} + \boldsymbol{w}_{\text{null}} \mid \boldsymbol{w}_{\text{null}} \in \operatorname{null}(\boldsymbol{X})\}$ we have

$$
\begin{aligned}
&= \boldsymbol{X}^{\dagger}\boldsymbol{y} + \operatorname*{arg\,min}_{\boldsymbol{w}_{\text{null}}\in\operatorname{null}(\boldsymbol{X})} \|\boldsymbol{w}_{\text{null}} - (\boldsymbol{w}_0^{(i)} - \boldsymbol{X}^{\dagger}\boldsymbol{y})\|_2\\
&= \boldsymbol{X}^{\dagger}\boldsymbol{y} + \operatorname{proj}_{\operatorname{null}(\boldsymbol{X})}\left(\boldsymbol{w}_0^{(i)} - \underbrace{\boldsymbol{X}^{\dagger}\boldsymbol{y}}_{\in\operatorname{range}(\boldsymbol{X}^{\top})}\right)
\end{aligned}
$$

where we used the characterization of an orthogonal projection onto a linear subspace as the (unique) closest point in the subspace. Finally, we use that the minimum norm solution is in the range space of the data and rewrite the projection in matrix form, s.t.

$$
= \boldsymbol{X}^{\dagger}\boldsymbol{y} + \boldsymbol{P}_{\text{null}}\boldsymbol{w}_0^{(i)}.
$$

Therefore the distribution over the parameters $\boldsymbol{w}_\infty^{(i)}$ of the ensemble members computed via (S)GD with initial parameters $\boldsymbol{w}_0 \sim \mathcal{N}\big(\boldsymbol{w}; \boldsymbol{\mu}_0, \boldsymbol{S}_0\boldsymbol{S}_0^{\top}\big)$ is given by

$$
q_{\text{Ens}}(\boldsymbol{w}) = \mathcal{N}\left(\boldsymbol{w}; \underbrace{\boldsymbol{X}^{\dagger}\boldsymbol{y} + \boldsymbol{P}_{\text{null}}\boldsymbol{\mu}_0}_{=\boldsymbol{\mu}_{\text{Ens}}}, \underbrace{\boldsymbol{P}_{\text{null}}\boldsymbol{S}_0}_{=\boldsymbol{S}_{\text{Ens}}}\boldsymbol{S}_0^{\top}\boldsymbol{P}_{\text{null}}^{\top}\right).
$$

Now the expected negative log-likelihood of the distribution over the parameters of the trained ensemble members $q_{\text{Ens}}(\boldsymbol{w})$ with hyperparameters $\boldsymbol{\theta}_{\text{Ens}} = (\boldsymbol{\mu}_{\text{Ens}}, \boldsymbol{S}_{\text{Ens}})$ is

$$
\bar{\ell}(\boldsymbol{\theta}_{\text{Ens}}) \overset{+c}{=} \frac{1}{2\sigma^2}\big(\|\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\mu}_{\text{Ens}}\|_2^2 + \operatorname{tr}(\boldsymbol{X}\boldsymbol{S}_{\text{Ens}}\boldsymbol{S}_{\text{Ens}}^{\top}\boldsymbol{X}^{\top})\big) = 0
$$

and therefore $\boldsymbol{\theta}_{\text{Ens}}$ is a minimizer of the expected log-likelihood. Further it holds that

$$
\boldsymbol{z} = \boldsymbol{V}_{\text{null}}^{\top}(\boldsymbol{P}_{\text{null}}\boldsymbol{\mu}_0) = \boldsymbol{V}_{\text{null}}^{\top}\boldsymbol{\mu}_0
$$

$$\boldsymbol{M} = \boldsymbol{V}_{\text{null}}^{\mathsf{T}}(\boldsymbol{P}_{\text{null}}\boldsymbol{S}_0)(\boldsymbol{P}_{\text{null}}\boldsymbol{S}_0)^{\mathsf{T}}\boldsymbol{V}_{\text{null}} = \boldsymbol{V}_{\text{null}}^{\mathsf{T}}\boldsymbol{S}_0\boldsymbol{S}_0^{\mathsf{T}}\boldsymbol{V}_{\text{null}} = \boldsymbol{V}_{\text{null}}^{\mathsf{T}}\boldsymbol{\Sigma}_0\boldsymbol{V}_{\text{null}}$$

and thus by Equation (S19), the distribution of the trained ensemble parameters minimizes the 2-Wasserstein distance to the prior distribution, i.e.

$$q_{\text{Ens}} = \underset{q(\boldsymbol{w})=\mathcal{N}(\boldsymbol{w};\boldsymbol{\mu},\boldsymbol{\Sigma})}{\arg\min} \mathrm{W}_2^2(q(\boldsymbol{w}), \mathcal{N}(\boldsymbol{w};\boldsymbol{\mu}_0,\boldsymbol{\Sigma}_0))\,.$$

Combining this with the characterization of the variational posterior in Theorem 1 proves the claim.

□

S1.2 Binary Classification of Linearly Separable Data

In this subsection we provide proofs of claims from Section 4.2. We begin with presenting some preliminary results from Soudry et al. [4] which will be used throughout the proof. Next, we will analyze the gradient flow of the expected loss. We extend the results for the gradient flow to gradient descent and derive the characterization of the implicit bias, completing the proof of Theorem 2.

Theorem 2 (Implicit Bias in Binary Classification)
Let $f_{\boldsymbol{w}}(\boldsymbol{x}) = \boldsymbol{x}^{\mathsf{T}}\boldsymbol{w}$ be an (overparametrized) linear model and define a Gaussian prior $p(\boldsymbol{w}) = \mathcal{N}(\boldsymbol{w};\boldsymbol{\mu}_0,\boldsymbol{S}_0\boldsymbol{S}_0^{\mathsf{T}})$. Assume a variational distribution $q_{\boldsymbol{\theta}}(\boldsymbol{w}) = \mathcal{N}(\boldsymbol{w};\boldsymbol{\mu},\boldsymbol{S}\boldsymbol{S}^{\mathsf{T}})$ over the weights $\boldsymbol{w} \in \mathbb{R}^P$ with variational parameters $\boldsymbol{\theta} = (\boldsymbol{\mu},\boldsymbol{S})$ such that $\boldsymbol{S} \in \mathbb{R}^{P\times R}$ and $R \leq P$. Assume we are using the exponential loss $\ell(u) = \exp(-u)$ and optimize the expected empirical loss $\bar{\ell}(\boldsymbol{\theta})$ via gradient descent initialized at the prior, i.e. $\boldsymbol{\theta}_0 = (\boldsymbol{\mu}_0,\boldsymbol{S}_0)$, with a sufficiently small learning rate η. Then for almost any dataset which is linearly separable (Assumption 1) and for which the support vectors span the data (Assumption 2), the rescaled gradient descent iterates (rGD)

$$\boldsymbol{\theta}_t^{\text{rGD}} = (\boldsymbol{\mu}_t^{\text{rGD}}, \boldsymbol{S}_t^{\text{rGD}}) = \left(\tfrac{1}{\log(t)}\boldsymbol{\mu}_t^{\text{GD}} + \boldsymbol{P}_{\text{null}(\boldsymbol{X})}\boldsymbol{\mu}_0, \boldsymbol{S}_t^{\text{GD}}\right) \tag{9}$$

converge to a limit point $\boldsymbol{\theta}_\star^{\text{rGD}} = \lim_{t\to\infty}\boldsymbol{\theta}_t^{\text{rGD}}$ for which it holds that

$$\boldsymbol{\theta}_\star^{\text{rGD}} \in \underset{\substack{\boldsymbol{\theta}=(\boldsymbol{\mu},\boldsymbol{S})\\ \text{s.t. } \boldsymbol{\theta}\in\Theta_\star}}{\arg\min} \mathrm{W}_2^2(q_{\boldsymbol{\theta}}, p)\,. \tag{10}$$

where the feasible set $\Theta_\star = \{(\boldsymbol{\mu},\boldsymbol{S}) \mid \boldsymbol{P}_{\text{range}(\boldsymbol{X}^{\mathsf{T}})}\boldsymbol{\mu} = \hat{\boldsymbol{\mu}} \quad \text{and} \quad \forall n : \mathrm{Var}_{q_{\boldsymbol{\theta}}}(f_{\boldsymbol{w}}(\boldsymbol{x}_n)) = 0\}$ consists of mean parameters which, if projected onto the training data, are equivalent to the L_2 max margin vector and covariance parameters such that there is no uncertainty at training data.

S1.2.1 Preliminaries

Recall that the expected loss is given by

$$\bar{\ell}(\boldsymbol{\theta}) = \textstyle\sum_{n=1}^N \mathbb{E}_{q_{\boldsymbol{\theta}}(\boldsymbol{w})}\left(\ell(y_n\boldsymbol{x}_n^{\mathsf{T}}\boldsymbol{w})\right), \tag{S22}$$

and specifically, for the exponential loss, we have

$$\bar{\ell}(\boldsymbol{\theta}) = \bar{\ell}(\boldsymbol{\mu},\boldsymbol{S}) = \textstyle\sum_{n=1}^N \exp\left(-\boldsymbol{x}_n^{\mathsf{T}}\boldsymbol{\mu} + \frac{1}{2}\boldsymbol{x}_n^{\mathsf{T}}\boldsymbol{S}\boldsymbol{S}^{\mathsf{T}}\boldsymbol{x}_n\right). \tag{S23}$$

Throughout these proofs, for any mean parameter iterate $\boldsymbol{\mu}_t$, we define the residual as

$$\boldsymbol{r}_t = \boldsymbol{\mu}_t - \hat{\boldsymbol{\mu}}\log t - \tilde{\boldsymbol{\mu}} \tag{S24}$$

where $\hat{\boldsymbol{\mu}}$ is the solution to the hard margin SVM, and $\tilde{\boldsymbol{\mu}}$ is the vector which satisfies

$$\forall n \in \mathcal{S} : \eta\exp\left(-\boldsymbol{x}_n^{\mathsf{T}}\tilde{\boldsymbol{\mu}}\right) = \alpha_n, \tag{S25}$$

where weights α_n are defined through the KKT conditions on the hard margin SVM problem, i.e.

$$\hat{\boldsymbol{\mu}} = \sum_{n\in\mathcal{S}} \alpha_n\boldsymbol{x}_n. \tag{S26}$$

In Lemma 12 (Appendix B) of Soudry et al. [4], it is shown that, for almost any dataset, there are no more than P support vectors and $\alpha_n \neq 0, \forall n \in \mathcal{S}$. Furthermore, we denote the minimum margin to a non-support vector as:

$$\kappa = \min_{n\notin\mathcal{S}} \boldsymbol{x}_n^{\mathsf{T}}\hat{\boldsymbol{\mu}} > 1. \tag{S27}$$

Finally, we define $\boldsymbol{P}_{\mathcal{S}} \in \mathbb{R}^{P\times P}$ as the orthogonal projection matrix to the subspace spanned by the support vectors, and $\bar{\boldsymbol{P}}_{\mathcal{S}} = \boldsymbol{I} - \boldsymbol{P}_{\mathcal{S}}$ as the complementary projection.

S1.2.2 Gradient Flow for the Expected Loss

Similar as in Soudry et al. [4], we begin by studying the gradient flow dynamics, i.e. taking the continuous time limit of gradient descent:

$$\dot{\boldsymbol{\theta}}_t = -\nabla \bar{\ell}(\boldsymbol{\theta}_t), \tag{S28}$$

which can be written componentwise as:

$$\dot{\boldsymbol{\mu}}_t = -\nabla_{\boldsymbol{\mu}} \bar{\ell}(\boldsymbol{\mu}_t, \boldsymbol{S}_t) = \sum_{n=1}^{N} \exp\left(-\boldsymbol{\mu}_t^\top \boldsymbol{x}_n + \frac{1}{2}\boldsymbol{x}_n^\top \boldsymbol{S}_t \boldsymbol{S}_t^\top \boldsymbol{x}_n\right)\boldsymbol{x}_n \tag{S29}$$

$$\dot{\boldsymbol{S}}_t = -\nabla_{\boldsymbol{S}} \bar{\ell}(\boldsymbol{\mu}_t, \boldsymbol{S}_t) = -\sum_{n=1}^{N} \exp\left(-\boldsymbol{\mu}_t^\top \boldsymbol{x}_n + \frac{1}{2}\boldsymbol{x}_n^\top \boldsymbol{S}_t \boldsymbol{S}_t^\top \boldsymbol{x}_n\right)\boldsymbol{x}_n \boldsymbol{x}_n^\top \boldsymbol{S}_t. \tag{S30}$$

We begin by showing that the total uncertainty, as measured by the Frobenius norm of the covariance factor, is bounded during the gradient flow dynamics. To that end, we derive the following dynamics:

$$\frac{d}{dt}\frac{1}{2}\|\boldsymbol{S}_t\|_F^2 = \operatorname{tr}(\boldsymbol{S}_t^\top \dot{\boldsymbol{S}}_t) = -\sum_{n=1}^{N} \exp\left(-\boldsymbol{\mu}_t^\top \boldsymbol{x}_n + \frac{1}{2}\boldsymbol{x}_n^\top \boldsymbol{S}_t \boldsymbol{S}_t^\top \boldsymbol{x}_n\right)\|\boldsymbol{x}_n^\top \boldsymbol{S}_t\|^2 \leq 0, \tag{S31}$$

and therefore

$$\|\boldsymbol{S}_t\|_F^2 \leq \|\boldsymbol{S}_0\|_F^2. \tag{S32}$$

Finally, by Cauchy-Schwarz inequality, we have that

$$\|\boldsymbol{S}_t \boldsymbol{S}_t^\top\|_F \leq \|\boldsymbol{S}_t\|_F^2 \leq \|\boldsymbol{S}_0\|_F^2. \tag{S33}$$

We continue by studying the convergence behavior of the mean parameter $\boldsymbol{\mu}_t$.

Mean parameter Our goal is to show that $\|\boldsymbol{r}_t\|$ is bounded. Equation (S24) implies that

$$\dot{\boldsymbol{r}}_t = \dot{\boldsymbol{\mu}}_t - \frac{1}{t}\hat{\boldsymbol{\mu}} = -\nabla_{\boldsymbol{\mu}} \bar{\ell}(\boldsymbol{\mu}_t, \boldsymbol{S}_t) - \frac{1}{t}\hat{\boldsymbol{\mu}}. \tag{S34}$$

This in turn implies that

$$\begin{aligned}
\frac{1}{2}\frac{d}{dt}\|\boldsymbol{r}_t\|^2 &= \dot{\boldsymbol{r}}_t^\top \boldsymbol{r}_t \\
&= \sum_{n=1}^{N} \exp\left(-\boldsymbol{\mu}_t^\top \boldsymbol{x}_n + \frac{1}{2}\boldsymbol{x}_n^\top \boldsymbol{S}_t \boldsymbol{S}_t^\top \boldsymbol{x}_n\right)\boldsymbol{x}_n^\top \boldsymbol{r}_t - \frac{1}{t}\hat{\boldsymbol{\mu}}^\top \boldsymbol{r}_t \\
&= \sum_{n\in\mathcal{S}} \exp\left(-\log(t)\hat{\boldsymbol{\mu}}^\top \boldsymbol{x}_n - \tilde{\boldsymbol{\mu}}^\top \boldsymbol{x}_n + \frac{1}{2}\boldsymbol{x}_n^\top \boldsymbol{S}_t \boldsymbol{S}_t^\top \boldsymbol{x}_n - \boldsymbol{x}_n^\top \boldsymbol{r}_t\right)\boldsymbol{x}_n^\top \boldsymbol{r}_t - \frac{1}{t}\hat{\boldsymbol{\mu}}^\top \boldsymbol{r}_t \\
&\quad + \sum_{n\notin\mathcal{S}} \exp\left(-\log(t)\hat{\boldsymbol{\mu}}^\top \boldsymbol{x}_n - \tilde{\boldsymbol{\mu}}^\top \boldsymbol{x}_n + \frac{1}{2}\boldsymbol{x}_n^\top \boldsymbol{S}_t \boldsymbol{S}_t^\top \boldsymbol{x}_n - \boldsymbol{x}_n^\top \boldsymbol{r}_t\right)\boldsymbol{x}_n^\top \boldsymbol{r}_t \\
&= \left[\frac{1}{t}\sum_{n\in\mathcal{S}} \exp\left(-\tilde{\boldsymbol{\mu}}^\top \boldsymbol{x}_n\right)\left(\exp\left(-\boldsymbol{x}_n^\top \boldsymbol{r}_t + \frac{1}{2}\boldsymbol{x}_n^\top \boldsymbol{S}_t \boldsymbol{S}_t^\top \boldsymbol{x}_n\right) - 1\right)\boldsymbol{x}_n^\top \boldsymbol{r}_t\right] \\
&\quad + \left[\sum_{n\notin\mathcal{S}} \left(\frac{1}{t}\right)^{\hat{\boldsymbol{\mu}}^\top \boldsymbol{x}_n} \exp\left(-\tilde{\boldsymbol{\mu}}^\top \boldsymbol{x}_n + \frac{1}{2}\boldsymbol{x}_n^\top \boldsymbol{S}_t \boldsymbol{S}_t^\top \boldsymbol{x}_n\right)\exp\left(-\boldsymbol{x}_n^\top \boldsymbol{r}_t\right)\boldsymbol{x}_n^\top \boldsymbol{r}_t\right].
\end{aligned} \tag{S35}$$

where in last line we used the fact that $\hat{\boldsymbol{\mu}}^\top \boldsymbol{x}_n = 1$ for $n \in \mathcal{S}$, and that $\sum_{n\in\mathcal{S}} \exp(-\boldsymbol{x}_n^\top \tilde{\boldsymbol{\mu}})\boldsymbol{x}_n = \hat{\boldsymbol{\mu}}$. We begin by examining the first bracket, studying three possible cases for each of the summands. First, note that if $\boldsymbol{x}_n^\top \boldsymbol{r}_t \leq 0$, then since $\frac{1}{2}\boldsymbol{x}_n^\top \boldsymbol{S}_t \boldsymbol{S}_t^\top \boldsymbol{x}_n \geq 0$, we have that

$$\left(\exp\left(-\boldsymbol{x}_n^\top \boldsymbol{r}_t + \frac{1}{2}\boldsymbol{x}_n^\top \boldsymbol{S}_t \boldsymbol{S}_t^\top \boldsymbol{x}_n\right) - 1\right)\boldsymbol{x}_n^\top \boldsymbol{r}_t \leq 0. \tag{S36}$$

Next, by defining $B := \|\boldsymbol{S}_0\|_F^2$, if $0 < \boldsymbol{x}_n^\mathsf{T}\boldsymbol{r}_t < \frac{B}{2}$, we have that

$$\left|\left(\exp\left(-\boldsymbol{x}_n^\mathsf{T}\boldsymbol{r}_t + \frac{1}{2}\boldsymbol{x}_n^\mathsf{T}\boldsymbol{S}_t\boldsymbol{S}_t^\mathsf{T}\boldsymbol{x}_n\right) - 1\right)\boldsymbol{x}_n^\mathsf{T}\boldsymbol{r}_t\right| < \left(\exp\left(\frac{B}{2}\right) - 1\right)\frac{B}{2}, \tag{S37}$$

and if $\boldsymbol{x}_n^\mathsf{T}\boldsymbol{r}_t \geq \frac{B}{2}$, we have that

$$\left(\exp\left(-\boldsymbol{x}_n^\mathsf{T}\boldsymbol{r}_t + \frac{1}{2}\boldsymbol{x}_n^\mathsf{T}\boldsymbol{S}_t\boldsymbol{S}_t^\mathsf{T}\boldsymbol{x}_n\right) - 1\right)\boldsymbol{x}_n^\mathsf{T}\boldsymbol{r}_t \leq 0. \tag{S38}$$

Finally, for arbitrary $\epsilon \geq \max\{B, 1\}$, if $|\boldsymbol{x}_n^\mathsf{T}\boldsymbol{r}_t| \geq \epsilon$, we have that

$$\left(\exp\left(-\boldsymbol{x}_n^\mathsf{T}\boldsymbol{r}_t + \frac{1}{2}\boldsymbol{x}_n^\mathsf{T}\boldsymbol{S}_t\boldsymbol{S}_t^\mathsf{T}\boldsymbol{x}_n\right) - 1\right)\boldsymbol{x}_n^\mathsf{T}\boldsymbol{r}_t \leq \left(\exp\left(-\frac{B}{2}\right) - 1\right)\epsilon < 0, \tag{S39}$$

Furthermore, let $\gamma_* = \min_{n\in\mathcal{S}} \tilde{\boldsymbol{\mu}}^\mathsf{T}\boldsymbol{x}_n$ and $\gamma^* = \max_{n\in\mathcal{S}} \tilde{\boldsymbol{\mu}}^\mathsf{T}\boldsymbol{x}_n$. Now, by taking $\epsilon \geq \max\{B, 1\}$ large enugh such that

$$\left|\exp(-\gamma^*)\left(\exp\left(-\frac{B}{2}\right) - 1\right)\epsilon\right| \geq |\mathcal{S}|\exp(-\gamma_*)\left(\exp\left(\frac{B}{2}\right) - 1\right)\frac{B}{2}, \tag{S40}$$

if there exists a support vector $n \in \mathcal{S}$ such that $|\boldsymbol{x}_n^\mathsf{T}\boldsymbol{r}_t| \geq \epsilon$, then

$$\frac{1}{t}\sum_{n\in\mathcal{S}} \exp\left(-\tilde{\boldsymbol{\mu}}^\mathsf{T}\boldsymbol{x}_n\right)\left(\exp\left(-\boldsymbol{x}_n^\mathsf{T}\boldsymbol{r}_t + \frac{1}{2}\boldsymbol{x}_n^\mathsf{T}\boldsymbol{S}_t\boldsymbol{S}_t^\mathsf{T}\boldsymbol{x}_n\right) - 1\right)\boldsymbol{x}_n^\mathsf{T}\boldsymbol{r}_t \leq 0. \tag{S41}$$

On the other hand, for the second bracket in Eq. (S35), note that for $n \notin \mathcal{S}$, we have that $\boldsymbol{x}_n^\mathsf{T}\hat{\boldsymbol{\mu}} \geq \kappa$, and hence

$$\begin{aligned}&\sum_{n\notin\mathcal{S}} \left(\frac{1}{t}\right)^{\hat{\boldsymbol{\mu}}^\mathsf{T}\boldsymbol{x}_n} \exp\left(-\tilde{\boldsymbol{\mu}}^\mathsf{T}\boldsymbol{x}_n + \frac{1}{2}\boldsymbol{x}_n^\mathsf{T}\boldsymbol{S}_t\boldsymbol{S}_t^\mathsf{T}\boldsymbol{x}_n\right)\exp\left(-\boldsymbol{x}_n^\mathsf{T}\boldsymbol{r}_t\right)\boldsymbol{x}_n^\mathsf{T}\boldsymbol{r}_t \\ &\leq \frac{1}{t^\kappa}\sum_{n\notin\mathcal{S}} \exp\left(-\tilde{\boldsymbol{\mu}}^\mathsf{T}\boldsymbol{x}_n + \frac{1}{2}\boldsymbol{x}_n^\mathsf{T}\boldsymbol{S}_t\boldsymbol{S}_t^\mathsf{T}\boldsymbol{x}_n\right) = \mathcal{O}\left(\frac{1}{t^\kappa}\right),\end{aligned} \tag{S42}$$

where in the last line we used that $ze^{-z} \leq 1, \forall z \in \mathbb{R}$ and fact that $\|\boldsymbol{S}_t\boldsymbol{S}_t^\mathsf{T}\|_F \leq \|\boldsymbol{S}_0\|_F^2 < \infty$.

We will now combine the results from above to show that the residual $\boldsymbol{r}_t$ is bounded in the following way: if there exists a support vector $n \in \mathcal{S}$ such that $|\boldsymbol{x}_n^\mathsf{T}\boldsymbol{r}_t| \geq \epsilon$ for big enough $\epsilon > 0$, then $\frac{1}{2}\frac{d}{dt}\|\boldsymbol{r}_t\|^2 = \mathcal{O}(t^{-\kappa})$. If such a support vector does not exist at time t, we will show that r_t is containted inside a compact set. To that end, if $\|\boldsymbol{P}_\mathcal{S}\boldsymbol{r}_t\| \geq \epsilon_1$, we have that

$$\max_{n\in\mathcal{S}} \left|\boldsymbol{x}_n^\mathsf{T}\boldsymbol{r}_t\right|^2 \geq \frac{1}{|\mathcal{S}|}\sum_{n\in\mathcal{S}} \left|\boldsymbol{x}_n^\mathsf{T}\boldsymbol{P}_\mathcal{S}\boldsymbol{r}_t\right|^2 = \frac{1}{|\mathcal{S}|}\left\|\boldsymbol{X}_\mathcal{S}^\mathsf{T}\boldsymbol{P}_\mathcal{S}\boldsymbol{r}_t\right\|^2 \geq \frac{1}{|\mathcal{S}|}\sigma_{\min}^2(\boldsymbol{X}_\mathcal{S})\epsilon_1^2, \tag{S43}$$

where in the first inequality we used the fact that $\boldsymbol{P}_\mathcal{S}^\mathsf{T}\boldsymbol{x}_n = \boldsymbol{x}_n$ for $n \in \mathcal{S}$. Hence by choosing ϵ_1 such that $\sigma_{\min}^2(\boldsymbol{X}_\mathcal{S})\epsilon_1^2/|\mathcal{S}| = \epsilon^2$, where the ϵ is chosen in Eq. (S40), we have that

$$\|\boldsymbol{P}_\mathcal{S}\boldsymbol{r}_t\| \geq \epsilon_1 \Rightarrow \frac{1}{2}\frac{d}{dt}\|\boldsymbol{r}_t\|^2 = \mathcal{O}\big(t^{-\kappa}\big). \tag{S44}$$

On the other hand, if $\|\boldsymbol{P}_\mathcal{S}\boldsymbol{r}_t\| \leq \epsilon_1$, recall that

$$\boldsymbol{r}_t = (\boldsymbol{\mu}_t - \boldsymbol{\mu}_0) + \boldsymbol{\mu}_0 - \hat{\boldsymbol{\mu}}\log t - \tilde{\boldsymbol{\mu}}, \tag{S45}$$

and since all updates to the mean parameter are in the space spanned by the support vectors (Assumption 2), we have that

$$\bar{\boldsymbol{P}}_\mathcal{S}\boldsymbol{r}_t = \bar{\boldsymbol{P}}_\mathcal{S}\boldsymbol{\mu}_0 - \bar{\boldsymbol{P}}_\mathcal{S}\tilde{\boldsymbol{\mu}}. \tag{S46}$$

We can now conclude that

$$\|\boldsymbol{P}_\mathcal{S}\boldsymbol{r}_t\| \leq \epsilon_1 \Rightarrow \|\boldsymbol{r}_t\| \leq \|\boldsymbol{P}_\mathcal{S}\boldsymbol{r}_t\| + \|\bar{\boldsymbol{P}}_\mathcal{S}\boldsymbol{r}_t\| \leq \epsilon_1 + \|\bar{\boldsymbol{P}}_\mathcal{S}\boldsymbol{\mu}_0\| + \|\bar{\boldsymbol{P}}_\mathcal{S}\tilde{\boldsymbol{\mu}}\| < \infty. \tag{S47}$$

Finally, combining the results from Eq. (S42) and Eq. (S47), recalling that $\kappa > 1$, we have that $\|\boldsymbol{r}_t\|$ is bounded for all $t > 0$. This completes the first part of the proof and shows that

$$\boldsymbol{\mu}_t = \hat{\boldsymbol{\mu}}\log t + \tilde{\boldsymbol{\mu}} + \boldsymbol{r}_t = \hat{\boldsymbol{\mu}}\log t + \mathcal{O}(1), \tag{S48}$$

and in particular

$$\lim_{t\to\infty}\frac{\boldsymbol{\mu}_t}{\|\boldsymbol{\mu}_t\|} = \frac{\hat{\boldsymbol{\mu}}}{\|\hat{\boldsymbol{\mu}}\|}. \tag{S49}$$

We proceed by showing that the limit covariance parameter vanishes in the span of the support vectors.

Covariance parameter We begin by plugging the definition of residual $\boldsymbol{r}_t$ (Equation (S24)) into the dynamics of $\boldsymbol{S}_t$:

$$\begin{aligned}
\dot{\boldsymbol{S}}_t = -\nabla_{\boldsymbol{S}}\bar{\ell}(\boldsymbol{\mu}_t, \boldsymbol{S}_t) &= -\sum_{n=1}^{N} \exp\left(-\boldsymbol{\mu}_t^{\mathsf{T}}\boldsymbol{x}_n + \frac{1}{2}\boldsymbol{x}_n^{\mathsf{T}}\boldsymbol{S}_t\boldsymbol{S}_t^{\mathsf{T}}\boldsymbol{x}_n\right)\boldsymbol{x}_n\boldsymbol{x}_n^{\mathsf{T}}\boldsymbol{S}_t \\
&= -\sum_{n\in\mathcal{S}} \frac{1}{t}\exp\left(-\tilde{\boldsymbol{\mu}}^{\mathsf{T}}\boldsymbol{x}_n - \boldsymbol{r}_t^{\mathsf{T}}\boldsymbol{x}_n\right)\exp\left(\frac{1}{2}\boldsymbol{x}_n^{\mathsf{T}}\boldsymbol{S}_t\boldsymbol{S}_t^{\mathsf{T}}\boldsymbol{x}_n\right)\boldsymbol{x}_n\boldsymbol{x}_n^{\mathsf{T}}\boldsymbol{S}_t \\
&\quad -\sum_{n\notin\mathcal{S}} \left(\frac{1}{t}\right)^{\boldsymbol{x}_n^{\mathsf{T}}\hat{\boldsymbol{\mu}}}\exp\left(-\tilde{\boldsymbol{\mu}}^{\mathsf{T}}\boldsymbol{x}_n - \boldsymbol{r}_t^{\mathsf{T}}\boldsymbol{x}_n\right)\exp\left(\frac{1}{2}\boldsymbol{x}_n^{\mathsf{T}}\boldsymbol{S}_t\boldsymbol{S}_t^{\mathsf{T}}\boldsymbol{x}_n\right)\boldsymbol{x}_n\boldsymbol{x}_n^{\mathsf{T}}\boldsymbol{S}_t,
\end{aligned} \tag{S50}$$

where we used that $\boldsymbol{x}_n^{\mathsf{T}}\hat{\boldsymbol{\mu}} = 1$ for $n \in \mathcal{S}$. Also, we know that $\|\boldsymbol{r}_t\|$ is bounded from the previous part. Hence, let

$$C := \min_{n\in[N]}\min_{t\geq 0}\exp\left(-\tilde{\boldsymbol{\mu}}^{\mathsf{T}}\boldsymbol{x}_n - \boldsymbol{r}_t^{\mathsf{T}}\boldsymbol{x}_n\right) > 0. \tag{S51}$$

Furthermore, let $\sigma_{\min}$ be the smallest non-zero eigenvalue of the matrix $\sum_{n\in\mathcal{S}}\boldsymbol{x}_n\boldsymbol{x}_n^{\mathsf{T}}$. Finally, we define

$$\Delta_t := \operatorname{tr}(\boldsymbol{P}_{\mathcal{S}}\boldsymbol{S}_t\boldsymbol{S}_t^{\mathsf{T}}\boldsymbol{P}_{\mathcal{S}})$$

to be the trace of the projection of the covariance parameter to the space of support vectors in $\mathcal{S}$. We compute its derivative over time and plug in the dynamics of $\boldsymbol{S}_t$:

$$\begin{aligned}
&\frac{1}{2}\frac{d}{dt}\Delta_t = \operatorname{tr}(\boldsymbol{P}_{\mathcal{S}}\dot{\boldsymbol{S}}_t\boldsymbol{S}_t^{\mathsf{T}}\boldsymbol{P}_{\mathcal{S}}) \\
&= -\frac{1}{t}\sum_{n\in\mathcal{S}}\exp\left(-\tilde{\boldsymbol{\mu}}^{\mathsf{T}}\boldsymbol{x}_n - \boldsymbol{r}_t^{\mathsf{T}}\boldsymbol{x}_n\right)\exp\left(\frac{1}{2}\boldsymbol{x}_n^{\mathsf{T}}\boldsymbol{S}_t\boldsymbol{S}_t^{\mathsf{T}}\boldsymbol{x}_n\right)\operatorname{tr}(\boldsymbol{P}_{\mathcal{S}}\boldsymbol{x}_n\boldsymbol{x}_n^{\mathsf{T}}\boldsymbol{S}_t\boldsymbol{S}_t^{\mathsf{T}}\boldsymbol{P}_{\mathcal{S}}) \\
&\quad -\sum_{n\notin\mathcal{S}}\left(\frac{1}{t}\right)^{\boldsymbol{x}_n^{\mathsf{T}}\hat{\boldsymbol{\mu}}}\exp\left(-\tilde{\boldsymbol{\mu}}^{\mathsf{T}}\boldsymbol{x}_n - \boldsymbol{r}_t^{\mathsf{T}}\boldsymbol{x}_n\right)\exp\left(\frac{1}{2}\boldsymbol{x}_n^{\mathsf{T}}\boldsymbol{S}_t\boldsymbol{S}_t^{\mathsf{T}}\boldsymbol{x}_n\right)\operatorname{tr}(\boldsymbol{P}_{\mathcal{S}}\boldsymbol{x}_n\boldsymbol{x}_n^{\mathsf{T}}\boldsymbol{S}_t\boldsymbol{S}_t^{\mathsf{T}}\boldsymbol{P}_{\mathcal{S}}) \\
&= -\frac{1}{t}\sum_{n\in\mathcal{S}}\exp\left(-\tilde{\boldsymbol{\mu}}^{\mathsf{T}}\boldsymbol{x}_n - \boldsymbol{r}_t^{\mathsf{T}}\boldsymbol{x}_n\right)\exp\left(\frac{1}{2}\boldsymbol{x}_n^{\mathsf{T}}\boldsymbol{S}_t\boldsymbol{S}_t^{\mathsf{T}}\boldsymbol{x}_n\right)\operatorname{tr}\big(\boldsymbol{P}_{\mathcal{S}}\boldsymbol{x}_n\boldsymbol{x}_n^{\mathsf{T}}\boldsymbol{S}_t\boldsymbol{S}_t^{\mathsf{T}}\boldsymbol{P}_{\mathcal{S}}\big) + \mathcal{O}\left(\frac{1}{t^{\kappa}}\right) \\
&\leq -\frac{C}{t}\sum_{n\in\mathcal{S}}\operatorname{tr}\big(\boldsymbol{P}_{\mathcal{S}}\boldsymbol{x}_n\boldsymbol{x}_n^{\mathsf{T}}\boldsymbol{S}_t\boldsymbol{S}_t^{\mathsf{T}}\boldsymbol{P}_{\mathcal{S}}\big) + \mathcal{O}\left(\frac{1}{t^{\kappa}}\right) \\
&= -\frac{C}{t}\operatorname{tr}\left(\boldsymbol{P}_{\mathcal{S}}\left(\sum_{n\in\mathcal{S}}\boldsymbol{x}_n\boldsymbol{x}_n^{\mathsf{T}}\right)\boldsymbol{S}_t\boldsymbol{S}_t^{\mathsf{T}}\boldsymbol{P}_{\mathcal{S}}\right) + \mathcal{O}\left(\frac{1}{t^{\kappa}}\right) \\
&\leq -\frac{C\sigma_{\min}}{t}\operatorname{tr}(\boldsymbol{P}_{\mathcal{S}}\boldsymbol{S}_t\boldsymbol{S}_t^{\mathsf{T}}\boldsymbol{P}_{\mathcal{S}}) + \mathcal{O}\left(\frac{1}{t^{\kappa}}\right) \\
&= -\frac{C\sigma_{\min}}{t}\Delta_t + \mathcal{O}\left(\frac{1}{t^{\kappa}}\right),
\end{aligned} \tag{S52}$$

where the first inequality follows from Eq. (S51), and the second from the definition of $\sigma_{\min}$. By Grönwall's lemma, we have that there exists a constant $K > 0$ such that, for some fixed $t_0 > 0$,

$$\Delta_t \leq \Delta_{t_0}\left(\tfrac{t}{t_0}\right)^{-2C\sigma_{\min}} + \frac{K}{2C\sigma_{\min} + \kappa - 1}t^{-(\kappa-1)}, \quad \forall t \geq t_0. \tag{S53}$$

Finally, since $|\mathcal{S}|\,C\sigma_{\min} > 0$ and $\kappa > 1$, we conclude that $\Delta_t \to 0$ as $t \to \infty$. This implies that the covariance parameter converges to zero in the span of the support vectors, i.e.

$$\forall n \in \mathcal{S}: \lim_{t\to\infty}\boldsymbol{x}_n^{\mathsf{T}}\boldsymbol{S}_t\boldsymbol{S}_t^{\mathsf{T}}\boldsymbol{x}_n = 0, \tag{S54}$$

as desired. □

S1.2.3 Complete Proof of Theorem 2

We will now extend the results for the gradient flow to gradient descent and then use these results to characterize the implicit bias of gradient descent as generalized variational inference.

Throughout this proof, let

$$\boldsymbol{A}_t = \sum_{n=1}^{N} \exp\left(-\boldsymbol{\mu}_t^\mathsf{T} \boldsymbol{x}_n + \frac{1}{2}\boldsymbol{x}_n^\mathsf{T} \boldsymbol{S}_t \boldsymbol{S}_t^\mathsf{T} \boldsymbol{x}_n\right) \boldsymbol{x}_n \boldsymbol{x}_n^\mathsf{T} \tag{S55}$$

be a positive definite matrix at iteration t. We begin the section with a few lemmata which will be used throughout the proof.

Lemma S2
Suppose that we start gradient descent from $(\boldsymbol{\mu}_0, \boldsymbol{S}_0)$. If $\eta < \lambda_{\max}(\boldsymbol{A}_0)^{-1}$, then for the gradient descent iterates

$$\boldsymbol{S}_{t+1} = \boldsymbol{S}_t - \eta \nabla_{\boldsymbol{S}} \bar{\ell}(\boldsymbol{\mu}_t, \boldsymbol{S}_t), \tag{S56}$$

we have that $\|\boldsymbol{S}_t\|_F \leq \|\boldsymbol{S}_0\|_F$ for all $t \geq 0$.

Proof. First, note that the gradient descent update for the covariance factor is given by

$$\boldsymbol{S}_{t+1} = \boldsymbol{S}_t(\boldsymbol{I} - \eta \boldsymbol{A}_t), \tag{S57}$$

and hence we have that

$$\|\boldsymbol{S}_{t+1}\|_F = \|\boldsymbol{S}_t(\boldsymbol{I} - \eta \boldsymbol{A}_t)\|_F \leq \|\boldsymbol{S}_t\|_F \|(\boldsymbol{I} - \eta \boldsymbol{A}_t)\|_2. \tag{S58}$$

Now, since $\eta \leq \lambda_{\max}(\boldsymbol{A}_0)^{-1} \leq \lambda_{\max}(\boldsymbol{A}_t)^{-1}$ for all $t \geq 0$ and noting that $\boldsymbol{A}_t \succeq 0$, we have that

$$\|(\boldsymbol{I} - \eta \boldsymbol{A}_t)\|_2 \leq 1, \tag{S59}$$

and therefore

$$\|\boldsymbol{S}_{t+1}\|_F \leq \|\boldsymbol{S}_t\|_F. \tag{S60}$$

Finally, we can conclude that $\|\boldsymbol{S}_t\|_F \leq \|\boldsymbol{S}_0\|_F$ for all $t \geq 0$, as required.

□

Lemma S3
Suppose that we start gradient descent from $(\boldsymbol{\mu}_0, \boldsymbol{S}_0)$. If $\eta < \lambda_{\max}(\boldsymbol{A}_0)^{-1}$, then for the gradient descent iterates

$$\boldsymbol{\mu}_{t+1} = \boldsymbol{\mu}_t - \eta \nabla_{\boldsymbol{\mu}} \bar{\ell}(\boldsymbol{\mu}_t, \boldsymbol{S}_t), \tag{S61}$$

we have that $\sum_{u=0}^{\infty} \|\nabla_{\boldsymbol{\mu}} \bar{\ell}(\boldsymbol{\mu}_u, \boldsymbol{S}_u)\|^2 < \infty$. Consequently, we also have that $\lim_{t\to\infty} \|\nabla_{\boldsymbol{\mu}} \bar{\ell}(\boldsymbol{\mu}_t, \boldsymbol{S}_t)\|^2 = 0$.

Proof. Note that our loss function is not globally smooth in $\boldsymbol{\mu}$. However, if we initialize at $(\boldsymbol{\mu}_0, \boldsymbol{S}_0)$, the gradient descent iterates with $\eta < \lambda_{\max}(\boldsymbol{A}_0)^{-1}$ maintain bounded local smoothness. The statement now follows directly from Lemma 10 in Soudry et al. [4]. □

Lemma S4
By choosing ϵ_1 as in Eq. (S44)*, if $\|\boldsymbol{P}_{\mathcal{S}} \boldsymbol{r}_t\| \geq \epsilon_1$, we have that*

$$(\boldsymbol{r}_{t+1} - \boldsymbol{r}_t)^\mathsf{T} \boldsymbol{r}_t \leq \mathcal{O}\left(\frac{1}{t^\kappa}\right) + \mathcal{O}\left(\frac{1}{t^2}\right)\|\boldsymbol{r}_t\|. \tag{S62}$$

If $\|\boldsymbol{P}_{\mathcal{S}} \boldsymbol{r}_t\| < \epsilon_1$, there exists a constant C such that

$$(\boldsymbol{r}_{t+1} - \boldsymbol{r}_t)^\mathsf{T} \boldsymbol{r}_t \leq C. \tag{S63}$$

Proof. We follow similar steps as in the gradient flow case. It holds that

$$\begin{aligned}
&(\boldsymbol{r}_{t+1}-\boldsymbol{r}_t)^{\mathsf{T}}\boldsymbol{r}_t \\
&= \left(-\eta\nabla_{\boldsymbol{\mu}}(\boldsymbol{\mu}_t,\boldsymbol{S}_t)-\hat{\boldsymbol{\mu}}\left(\log(t+1)-\log(t)\right)\right)^{\mathsf{T}}\boldsymbol{r}_t \\
&= \eta\sum_{n=1}^{N}\exp\left(-\boldsymbol{\mu}_t^{\mathsf{T}}\boldsymbol{x}_n+\frac{1}{2}\boldsymbol{x}_n^{\mathsf{T}}\boldsymbol{S}_t\boldsymbol{S}_t^{\mathsf{T}}\boldsymbol{x}_n\right)\boldsymbol{x}_n^{\mathsf{T}}\boldsymbol{r}_t-\hat{\boldsymbol{\mu}}^{\mathsf{T}}\boldsymbol{r}_t\log(1+t^{-1}) \\
&= \hat{\boldsymbol{\mu}}^{\mathsf{T}}\boldsymbol{r}_t(t^{-1}-\log(1+t^{-1}))+\eta\sum_{n\notin\mathcal{S}}\exp\left(-\boldsymbol{\mu}_t^{\mathsf{T}}\boldsymbol{x}_n+\frac{1}{2}\boldsymbol{x}_n^{\mathsf{T}}\boldsymbol{S}_t\boldsymbol{S}_t^{\mathsf{T}}\boldsymbol{x}_n\right)\boldsymbol{x}_n^{\mathsf{T}}\boldsymbol{r}_t \\
&+\eta\sum_{n\in\mathcal{S}}\left[-\frac{1}{t}\exp\left(-\tilde{\boldsymbol{\mu}}^{\mathsf{T}}\boldsymbol{x}_n\right)+\exp\left(-\boldsymbol{\mu}_t^{\mathsf{T}}\boldsymbol{x}_n+\frac{1}{2}\boldsymbol{x}_n^{\mathsf{T}}\boldsymbol{S}_t\boldsymbol{S}_t^{\mathsf{T}}\boldsymbol{x}_n\right)\right]\boldsymbol{x}_n^{\mathsf{T}}\boldsymbol{r}_t,
\end{aligned} \tag{S64}$$

where in the last equality we used Equation (S26) to expand $\hat{\boldsymbol{\mu}}^{\mathsf{T}}\boldsymbol{r}_t$. Furthermore, we can bound all four terms as follows, beginning with the first term:

$$\hat{\boldsymbol{\mu}}^{\mathsf{T}}\boldsymbol{r}_t(t^{-1}-\log(1+t^{-1}))\leq\|\boldsymbol{r}_t\|\mathcal{O}\left(\frac{1}{t^2}\right), \tag{S65}$$

where we used that $\log(1+t^{-1})=t^{-1}+\mathcal{O}(t^{-2})$. For the second term, using the same argument as in Equation (S42), we derive that

$$\eta\sum_{n\notin\mathcal{S}}\exp\left(-\boldsymbol{\mu}_t^{\mathsf{T}}\boldsymbol{x}_n+\frac{1}{2}\boldsymbol{x}_n^{\mathsf{T}}\boldsymbol{S}_t\boldsymbol{S}_t^{\mathsf{T}}\boldsymbol{x}_n\right)\boldsymbol{x}_n^{\mathsf{T}}\boldsymbol{r}_t\leq\mathcal{O}\left(\frac{1}{t^{\kappa}}\right). \tag{S66}$$

For the third item, from Eq. (S41) and Eq. (S43), we have that $\|\boldsymbol{P}_{\mathcal{S}}\boldsymbol{r}_t\|\geq\epsilon_1$ implies that

$$\eta\sum_{n\in\mathcal{S}}\left[-\frac{1}{t}\exp\left(-\tilde{\boldsymbol{\mu}}^{\mathsf{T}}\boldsymbol{x}_n\right)+\exp\left(-\boldsymbol{\mu}_t^{\mathsf{T}}\boldsymbol{x}_n+\frac{1}{2}\boldsymbol{x}_n^{\mathsf{T}}\boldsymbol{S}_t\boldsymbol{S}_t^{\mathsf{T}}\boldsymbol{x}_n\right)\right]\boldsymbol{x}_n^{\mathsf{T}}\boldsymbol{r}_t\leq 0. \tag{S67}$$

The first result follows from combining the above three inequalities.

Next, if $\|\boldsymbol{P}_{\mathcal{S}}\boldsymbol{r}_t\|<\epsilon_1$, by defining $B:=\|\boldsymbol{S}_0\|_F^2$, following the steps in Eq. (S37), we have that

$$\eta\sum_{n\notin\mathcal{S}}\exp\left(-\boldsymbol{\mu}_t^{\mathsf{T}}\boldsymbol{x}_n+\frac{1}{2}\boldsymbol{x}_n^{\mathsf{T}}\boldsymbol{S}_t\boldsymbol{S}_t^{\mathsf{T}}\boldsymbol{x}_n\right)\boldsymbol{x}_n^{\mathsf{T}}\boldsymbol{r}_t\leq\eta|\mathcal{S}|\left(\exp\left(\frac{B}{2}\right)-1\right)\frac{B}{2}, \tag{S68}$$

and hence, combining this with Assumption 2 which implies that $\boldsymbol{r}_t$ is bounded as in Eq. (S47), one can find a constant C such that

$$\left(\boldsymbol{r}_{t+1}-\boldsymbol{r}_t\right)^{\mathsf{T}}\boldsymbol{r}_t\leq C. \tag{S69}$$

□

Proof of Theorem 2

Proof. As in the simple version of the proof, we begin by considering the convergence behavior of the mean parameter $\boldsymbol{\mu}_t$.

Mean parameter Our goal is again to show that $\|\boldsymbol{r}_t\|$ is bounded. To that end, we will provide an upper bound to the following equation

$$\|\boldsymbol{r}_{t+1}\|^2=\|\boldsymbol{r}_{t+1}-\boldsymbol{r}_t\|^2+2\left(\boldsymbol{r}_{t+1}-\boldsymbol{r}_t\right)^{\mathsf{T}}\boldsymbol{r}_t+\|\boldsymbol{r}_t\|^2 \tag{S70}$$

First, consider the first term in the above equation:

$$\begin{aligned}
&\|\boldsymbol{r}_{t+1}-\boldsymbol{r}_t\|^2 \\
&=\|\boldsymbol{\mu}_{t+1}-\hat{\boldsymbol{\mu}}\log(t+1)-\tilde{\boldsymbol{\mu}}-\boldsymbol{\mu}_t+\hat{\boldsymbol{\mu}}\log(t)+\tilde{\boldsymbol{\mu}}\|^2 \\
&=\|-\eta\nabla_{\boldsymbol{\mu}}\bar{\ell}(\boldsymbol{\mu}_t,\boldsymbol{S}_t)-\hat{\boldsymbol{\mu}}\log(1+t^{-1})]\|^2 \\
&\leq 2\Big[\eta^2\|\nabla_{\boldsymbol{\mu}}\bar{\ell}(\boldsymbol{\mu}_t,\boldsymbol{S}_t)\|^2+\|\hat{\boldsymbol{\mu}}\|^2\log^2(1+t^{-1})\Big] \\
&\leq 2\Big[\eta^2\|\nabla_{\boldsymbol{\mu}}\bar{\ell}(\boldsymbol{\mu}_t,\boldsymbol{S}_t)\|^2+\|\hat{\boldsymbol{\mu}}\|^2t^{-2}\Big]
\end{aligned} \tag{S71}$$

where in the first inequality we used the standard inequality that $(x+y)^2 \leq 2(x^2+y^2)$, and in the second inequality we used the fact that $\log(1+x) \leq x$ for $x \geq 0$. Now, from Lemma S3 and the fact that t^{-2} is summable, we conclude that there exists $C_1 < \infty$ such that

$$\sum_{t=1}^{\infty} \|\boldsymbol{r}_{t+1} - \boldsymbol{r}_t\|^2 \leq C_1 < \infty. \tag{S72}$$

Next, for the second term, recall that in Lemma S4 we showed that if $\|\boldsymbol{P}_{\mathcal{S}} \boldsymbol{r}_t\| \geq \epsilon_1$, then, for some constants $C_2, C_3 < \infty$, we have that, eventually

$$(\boldsymbol{r}_{t+1} - \boldsymbol{r}_t)^{\top} \boldsymbol{r}_t \leq C_2 \frac{1}{t^{\kappa}} + C_3 \frac{1}{t^2} \|\boldsymbol{r}_t\|, \tag{S73}$$

and that if $\|\boldsymbol{P}_{\mathcal{S}} \boldsymbol{r}_t\| < \epsilon_1$, then there exists a constant $C_4 < \infty$ such that

$$(\boldsymbol{r}_{t+1} - \boldsymbol{r}_t)^{\top} \boldsymbol{r}_t \leq C_4. \tag{S74}$$

We will show that when $\|\boldsymbol{P}_{\mathcal{S}} \boldsymbol{r}_t\| < \epsilon_1$, the residual $\boldsymbol{r}_t$ is contained in a compact set, and when $\|\boldsymbol{P}_{\mathcal{S}} \boldsymbol{r}_t\| \geq \epsilon_1$, the residual $\boldsymbol{r}_t$ can't escape to infinity. We now formally show this claim.

Let S_1 be the frst time such that $\|\boldsymbol{P}_{\mathcal{S}} \boldsymbol{r}_t\| \geq \epsilon_1$, if such a time does not exist, we are done since the support vectors span the data and hence $\|\boldsymbol{r}_t\|$ is bounded. Now, let T_1 be the first time after S_1 such that $\|\boldsymbol{P}_{\mathcal{S}} \boldsymbol{r}_t\| < \epsilon_1$, where we allow $T_1 = \infty$ if such a time does not exist. Continuing in this manner, we define the sequences $S_1 < T_1 < S_2 < T_2 < \ldots$, where we allow $T_i = \infty$ for some i.

We prooced by showing that $\|\boldsymbol{r}_t\|$ is uniformly bounded on each of the intervals $[S_i, T_i)$. To that end, note that for $t \in [S_i, T_i)$, we have that

$$\|\boldsymbol{r}_{t+1}\|^2 - \|\boldsymbol{r}_t\|^2 \leq 2C_2 \frac{1}{t^{\kappa}} + 2C_3 \frac{1}{t^2} \|\boldsymbol{r}_t\| + \|\boldsymbol{r}_{t+1} - \boldsymbol{r}_t\|^2, \tag{S75}$$

and hence, using the fact that $\kappa > 1$, by the discrete version of Grönwall's lemma, that

$$\max_{t \in [S_i, T_i)} (\|\boldsymbol{r}_t\|^2 - \|\boldsymbol{r}_{S_i}\|^2) \leq K, \tag{S76}$$

for some constant $K < \infty$ independent of i. Furthemore, we also know from Eq. (S74) that

$$\|\boldsymbol{r}_{S_i}\| \leq \epsilon_1 + 2C_4 + \|\boldsymbol{r}_{S_i} - \boldsymbol{r}_{S_i - 1}\|^2 \leq \epsilon_1 + 2C_4 + \max_{t \geq 0} \|\boldsymbol{r}_{t+1} - \boldsymbol{r}_t\|^2 < \infty, \tag{S77}$$

showing that the first jump outisde the ϵ_1-ball is bounded. Combining the two results, we conclude that $\|\boldsymbol{r}_t\|$ is uniformly bounded on each of the intervals $[S_i, T_i)$.

Finally, by noting that the support vectors span the data, we have that $\|\boldsymbol{r}_t\|$ is uniformly bounded on each of the intervals $[T_i, S_{i+1})$. Combining the two results, we conclude that $\|\boldsymbol{r}_t\|$ is uniformly bounded for all $t \geq 0$ and hence we have that

$$\lim_{t \to \infty} \frac{\boldsymbol{\mu}_t}{\|\boldsymbol{\mu}_t\|} = \frac{\hat{\boldsymbol{\mu}}}{\|\hat{\boldsymbol{\mu}}\|} \tag{S78}$$

and the following lemma.

Lemma S5
For the mean parameter $\boldsymbol{\mu}_t$, we have that

$$\boldsymbol{\mu}_t = \log(t) \hat{\boldsymbol{\mu}} + \mathcal{O}(1). \tag{S79}$$

Proof. This follows immediately from the definition of the residual in Equation (S24):

$$\boldsymbol{\mu}_t = \hat{\boldsymbol{\mu}} \log t + \boldsymbol{r}_t + \tilde{\boldsymbol{\mu}}_t,$$

and the fact that $\boldsymbol{r}_t$ and $\tilde{\boldsymbol{\mu}}_t$ are bounded as we showed above. □

We continue with the analysis of the covariance parameter over optimization iterations.

Covariance parameter As before, let $\Delta_t = \operatorname{tr}(\boldsymbol{P}_{\mathcal{S}}\boldsymbol{S}_t\boldsymbol{S}_t^{\mathsf{T}}\boldsymbol{P}_{\mathcal{S}})$ be the trace of the projection of the covariance parameter on the space of support vectors in $\mathcal{S}$. By following the ideas from the gradient flow case, we have the following dynamics:

$$\begin{aligned}
\Delta_{t+1} &= \operatorname{tr}(\boldsymbol{P}_{\mathcal{S}}\left(\boldsymbol{I}-\eta\boldsymbol{A}_t\right)\boldsymbol{S}_t\boldsymbol{S}_t^{\mathsf{T}}\left(\boldsymbol{I}-\eta\boldsymbol{A}_t\right)^{\mathsf{T}}\boldsymbol{P}_{\mathcal{S}}) \\
&= \operatorname{tr}(\boldsymbol{P}_{\mathcal{S}}\boldsymbol{S}_t\boldsymbol{S}_t^{\mathsf{T}}\boldsymbol{P}_{\mathcal{S}}) - 2\eta\operatorname{tr}(\boldsymbol{P}_{\mathcal{S}}\boldsymbol{S}_t\boldsymbol{S}_t^{\mathsf{T}}\boldsymbol{A}_t\boldsymbol{P}_{\mathcal{S}}) + \eta^2\operatorname{tr}(\boldsymbol{P}_{\mathcal{S}}\boldsymbol{A}_t\boldsymbol{S}_t\boldsymbol{S}_t^{\mathsf{T}}\boldsymbol{A}_t\boldsymbol{P}_{\mathcal{S}}) \\
&\leq \Delta_t - \frac{2\eta}{t}C\sigma_{\min}\operatorname{tr}(\boldsymbol{P}_{\mathcal{S}}\boldsymbol{S}_t\boldsymbol{S}_t^{\mathsf{T}}\boldsymbol{P}_{\mathcal{S}}) + \mathcal{O}\left(\frac{1}{t^{\kappa}}\right) + \mathcal{O}\left(\frac{1}{t^2}\right) \\
&= \Delta_t - \frac{2\eta}{t}C\sigma_{\min}\Delta_t + \mathcal{O}\left(\frac{1}{t^{\kappa}}\right) + \mathcal{O}\left(\frac{1}{t^2}\right),
\end{aligned} \tag{S80}$$

where we used the same arguments as in Equation (S52) to derive the last inequality, in addition to noting that $\lambda_{\max}(\boldsymbol{A}_t^2) \leq \mathcal{O}\big(\frac{1}{t^2}\big)$ in order to bound the last term. Hence, we can write

$$\Delta_{t+1} - \Delta_t \leq -\frac{2\eta}{t}C\sigma_{\min}\Delta_t + \mathcal{O}\left(\frac{1}{t^{\kappa}}\right) + \mathcal{O}\left(\frac{1}{t^2}\right). \tag{S81}$$

Again, by the discrete version of Grönwall's lemma, we derive the equivalent result to Eq. (S53). Now, noting that $\sum_t \frac{1}{t}$ diverges, the fact that $\kappa > 1$ and $\eta C\sigma_{\min} > 0$, we conclude that Δ_t converges to zero. This implies that the covariance parameter converges to zero in the span of the support vectors, i.e.

$$\forall n \in \mathcal{S} : \lim_{t\to\infty} \boldsymbol{x}_n^{\mathsf{T}}\boldsymbol{S}_t\boldsymbol{S}_t^{\mathsf{T}}\boldsymbol{x}_n = 0, \tag{S82}$$

as desired.

Characterization as Generalized Variational Inference As a final step we need to show that the solution identified by gradient descent if appropriately transformed identifies the minimum 2-Wasserstein solution in the feasible set. Define the feasible set

$$\begin{aligned}
\Theta_\star &= \{(\boldsymbol{\mu}, \boldsymbol{S}) \mid \boldsymbol{P}_{\mathcal{S}}\boldsymbol{\mu} = \hat{\boldsymbol{\mu}} \quad \text{and} \quad \forall n \in \mathcal{S} : \operatorname{Var}_{q_{\boldsymbol{\theta}}}(f_{\boldsymbol{w}}(\boldsymbol{x}_n)) = 0\} && \text{(S83)} \\
&= \{(\boldsymbol{\mu}, \boldsymbol{S}) \mid \boldsymbol{P}_{\mathcal{S}}\boldsymbol{\mu} = \hat{\boldsymbol{\mu}} \quad \text{and} \quad \forall n \in \mathcal{S} : \boldsymbol{x}_n^{\mathsf{T}}\boldsymbol{S}\boldsymbol{S}^{\mathsf{T}}\boldsymbol{x}_n = 0\} && \text{(S84)}
\end{aligned}$$

and the variational parameters identified by rescaled gradient descent as

$$\boldsymbol{\theta}_\star^{\text{rGD}} = \lim_{t\to\infty}\boldsymbol{\theta}_t^{\text{rGD}} = \lim_{t\to\infty}\left(\frac{1}{\log(t)}\boldsymbol{\mu}_t + \boldsymbol{P}_{\text{null}(\boldsymbol{X})}\boldsymbol{\mu}_0, \boldsymbol{S}_t\right). \tag{S85}$$

It holds by Lemma S5 that

$$\boldsymbol{P}_{\mathcal{S}}\boldsymbol{\mu}_\star^{\text{rGD}} = \boldsymbol{P}_{\mathcal{S}}\left(\lim_{t\to\infty}\frac{1}{\log(t)}\boldsymbol{\mu}_t\right) + \boldsymbol{0} = \boldsymbol{P}_{\mathcal{S}}\hat{\boldsymbol{\mu}} = \hat{\boldsymbol{\mu}} \tag{S86}$$

and additionally by Equation (S82) we have for all $n \in \mathcal{S}$ that

$$\boldsymbol{x}_n^{\mathsf{T}}\boldsymbol{S}_\star^{\text{rGD}}(\boldsymbol{S}_\star^{\text{rGD}})^{\mathsf{T}}\boldsymbol{x}_n = \lim_{t\to\infty}\boldsymbol{x}_n^{\mathsf{T}}\boldsymbol{S}_t(\boldsymbol{S}_t)^{\mathsf{T}}\boldsymbol{x}_n = 0. \tag{S87}$$

Therefore the limit point $\boldsymbol{\theta}_\star^{\text{rGD}}$ of rescaled gradient descent is in the feasible set. It remains to show that it is also a minimizer of the 2-Wasserstein distance to the prior / initialization. We will first show a more general result that does not require Assumption 2.

To that end define $\begin{pmatrix}\boldsymbol{V}_{\mathcal{S}} & \boldsymbol{V}_{\boldsymbol{X}\perp\mathcal{S}} & \boldsymbol{V}_{\text{null}(\boldsymbol{X})}\end{pmatrix} \in \mathbb{R}^{P\times P}$ where $\boldsymbol{V}_{\mathcal{S}} \in \mathbb{R}^{P\times P_{\mathcal{S}}}$ is an orthonormal basis of the span of the support vectors $\operatorname{range}(\boldsymbol{X}_{\mathcal{S}}^{\mathsf{T}})$, $\boldsymbol{V}_{\boldsymbol{X}\perp\mathcal{S}} \in \mathbb{R}^{P\times(N-P_{\mathcal{S}})}$ an orthonormal basis of its orthogonal complement in $\operatorname{range}(\boldsymbol{X}^{\mathsf{T}})$ and $\boldsymbol{V}_{\text{null}(\boldsymbol{X})} \in \mathbb{R}^{P\times(P-N)}$ the corresponding orthonormal basis of the null space $\operatorname{null}(\boldsymbol{X})$ of the data. Let $\boldsymbol{V} = \begin{pmatrix}\boldsymbol{V}_{\mathcal{S}} & \boldsymbol{V}_{\text{null}(\boldsymbol{X})}\end{pmatrix} \in \mathbb{R}^{P\times(P-N+P_{\mathcal{S}})}$ and define the projected variational distribution and prior onto the span of the support vectors and the null space of the data as

$$q_{\boldsymbol{\theta}}^{\text{proj}}(\tilde{\boldsymbol{w}}) = \mathcal{N}\big(\tilde{\boldsymbol{w}}; \boldsymbol{P}_{\boldsymbol{V}}\boldsymbol{\mu}, \boldsymbol{P}_{\boldsymbol{V}}\boldsymbol{\Sigma}\boldsymbol{P}_{\boldsymbol{V}}^{\mathsf{T}}\big) = \mathcal{N}\Big(\tilde{\boldsymbol{w}}; \tilde{\boldsymbol{\mu}}, \tilde{\boldsymbol{\Sigma}}\Big) \tag{S88}$$

$$p^{\text{proj}}(\tilde{\boldsymbol{w}}) = \mathcal{N}\big(\tilde{\boldsymbol{w}}; \boldsymbol{P}_{\boldsymbol{V}}\boldsymbol{\mu}_0, \boldsymbol{P}_{\boldsymbol{V}}\boldsymbol{\Sigma}_0\boldsymbol{P}_{\boldsymbol{V}}^{\mathsf{T}}\big) = \mathcal{N}\Big(\tilde{\boldsymbol{w}}; \tilde{\boldsymbol{\mu}}_0, \tilde{\boldsymbol{\Sigma}}_0\Big) \tag{S89}$$

where $\tilde{\boldsymbol{w}} \in \mathbb{R}^{P-N+P_{\mathcal{S}}}$. Now earlier we showed that the limit point of rescaled gradient descent is in the feasible set, defined in Equation (S85), and thus the same holds for the projected limit point of rescaled gradient descent, i.e.

$$(\tilde{\boldsymbol{\mu}}_\star^{\text{rGD}}, \tilde{\boldsymbol{S}}_\star^{\text{rGD}}) \in \Theta_\star \tag{S90}$$

in particular

$$\boldsymbol{P}_{\mathcal{S}} \tilde{\boldsymbol{\mu}}_\star^{\text{rGD}} = \boldsymbol{P}_{\mathcal{S}} \boldsymbol{\mu}_\star^{\text{rGD}} = \hat{\boldsymbol{\mu}}, \tag{S91}$$

$$\forall n \in \mathcal{S}: \quad \boldsymbol{x}_n^\top \tilde{\boldsymbol{S}}_\star^{\text{rGD}} (\tilde{\boldsymbol{S}}_\star^{\text{rGD}})^\top \boldsymbol{x}_n = \boldsymbol{x}_n^\top \boldsymbol{S}_\star^{\text{rGD}} (\boldsymbol{S}_\star^{\text{rGD}})^\top \boldsymbol{x}_n = 0. \tag{S92}$$

Therefore we have for all $n \in \mathcal{S}$ that

$$0 = \boldsymbol{x}_n^\top \tilde{\boldsymbol{S}}_\star^{\text{rGD}} (\tilde{\boldsymbol{S}}_\star^{\text{rGD}})^\top \boldsymbol{x}_n = \|(\tilde{\boldsymbol{S}}_\star^{\text{rGD}})^\top \boldsymbol{x}_n\|_2^2 \iff (\tilde{\boldsymbol{S}}_\star^{\text{rGD}})^\top \boldsymbol{x}_n = \mathbf{0} \tag{S93}$$

$$\iff (\tilde{\boldsymbol{S}}_\star^{\text{rGD}})^\top \boldsymbol{V}_{\mathcal{S}} = \mathbf{0} \tag{S94}$$

and thus $\boldsymbol{V}_{\mathcal{S}}^\top \tilde{\boldsymbol{S}}_\star^{\text{rGD}} (\tilde{\boldsymbol{S}}_\star^{\text{rGD}})^\top \boldsymbol{V}_{\mathcal{S}} = \mathbf{0}$. Therefore by Lemma S1 it holds for the squared 2-Wasserstein distance between the projected limit point of rescaled gradient descent and the projected prior that

$$\begin{aligned}
\mathrm{W}_2^2\Big(q_{\boldsymbol{\theta}_*}^{\text{proj}}, p^{\text{proj}}\Big) &\overset{\pm c}{=} \|\boldsymbol{V}_{\mathcal{S}}^\top \tilde{\boldsymbol{\mu}} - \boldsymbol{V}_{\mathcal{S}}^\top \tilde{\boldsymbol{\mu}}_0\|_2^2 + \mathrm{W}_2^2\Big(\mathcal{N}\Big(\boldsymbol{V}_{\text{null}}^\top \tilde{\boldsymbol{\mu}}, \boldsymbol{V}_{\text{null}}^\top \tilde{\boldsymbol{\Sigma}} \boldsymbol{V}_{\text{null}}\Big), \mathcal{N}\Big(\boldsymbol{V}_{\text{null}}^\top \tilde{\boldsymbol{\mu}}_0, \boldsymbol{V}_{\text{null}}^\top \tilde{\boldsymbol{\Sigma}}_0 \boldsymbol{V}_{\text{null}}\Big)\Big) \\
&= \left\| \begin{pmatrix} \boldsymbol{V}_{\mathcal{S}}^\top \tilde{\boldsymbol{\mu}} - \boldsymbol{V}_{\mathcal{S}}^\top \tilde{\boldsymbol{\mu}}_0 \\ \mathbf{0} \end{pmatrix} \right\|_2^2 + \mathrm{W}_2^2\Big(\mathcal{N}\Big(\boldsymbol{V}_{\text{null}}^\top \tilde{\boldsymbol{\mu}}, \boldsymbol{V}_{\text{null}}^\top \tilde{\boldsymbol{\Sigma}} \boldsymbol{V}_{\text{null}}\Big), \mathcal{N}\Big(\boldsymbol{V}_{\text{null}}^\top \tilde{\boldsymbol{\mu}}_0, \boldsymbol{V}_{\text{null}}^\top \tilde{\boldsymbol{\Sigma}}_0 \boldsymbol{V}_{\text{null}}\Big)\Big) \\
&= \left\| \boldsymbol{V} \begin{pmatrix} \boldsymbol{V}_{\mathcal{S}}^\top \tilde{\boldsymbol{\mu}} - \boldsymbol{V}_{\mathcal{S}}^\top \tilde{\boldsymbol{\mu}}_0 \\ \mathbf{0} \end{pmatrix} \right\|_2^2 + \mathrm{W}_2^2\Big(\mathcal{N}\Big(\boldsymbol{V}_{\text{null}}^\top \tilde{\boldsymbol{\mu}}, \boldsymbol{V}_{\text{null}}^\top \tilde{\boldsymbol{\Sigma}} \boldsymbol{V}_{\text{null}}\Big), \mathcal{N}\Big(\boldsymbol{V}_{\text{null}}^\top \tilde{\boldsymbol{\mu}}_0, \boldsymbol{V}_{\text{null}}^\top \tilde{\boldsymbol{\Sigma}}_0 \boldsymbol{V}_{\text{null}}\Big)\Big) \\
&= \|\boldsymbol{P}_{\mathcal{S}} \tilde{\boldsymbol{\mu}} - \boldsymbol{P}_{\mathcal{S}} \tilde{\boldsymbol{\mu}}_0\|_2^2 + \mathrm{W}_2^2\Big(\mathcal{N}\Big(\boldsymbol{V}_{\text{null}}^\top \tilde{\boldsymbol{\mu}}, \boldsymbol{V}_{\text{null}}^\top \tilde{\boldsymbol{\Sigma}} \boldsymbol{V}_{\text{null}}\Big), \mathcal{N}\Big(\boldsymbol{V}_{\text{null}}^\top \tilde{\boldsymbol{\mu}}_0, \boldsymbol{V}_{\text{null}}^\top \tilde{\boldsymbol{\Sigma}}_0 \boldsymbol{V}_{\text{null}}\Big)\Big) \\
&= \|\hat{\boldsymbol{\mu}} - \boldsymbol{P}_{\mathcal{S}} \tilde{\boldsymbol{\mu}}_0\|_2^2 + \mathrm{W}_2^2\Big(\mathcal{N}\Big(\boldsymbol{V}_{\text{null}}^\top \tilde{\boldsymbol{\mu}}, \boldsymbol{V}_{\text{null}}^\top \tilde{\boldsymbol{\Sigma}} \boldsymbol{V}_{\text{null}}\Big), \mathcal{N}\Big(\boldsymbol{V}_{\text{null}}^\top \tilde{\boldsymbol{\mu}}_0, \boldsymbol{V}_{\text{null}}^\top \tilde{\boldsymbol{\Sigma}}_0 \boldsymbol{V}_{\text{null}}\Big)\Big) \\
&\overset{\pm c}{=} \mathrm{W}_2^2\Big(\mathcal{N}\Big(\boldsymbol{V}_{\text{null}}^\top \tilde{\boldsymbol{\mu}}, \boldsymbol{V}_{\text{null}}^\top \tilde{\boldsymbol{\Sigma}} \boldsymbol{V}_{\text{null}}\Big), \mathcal{N}\Big(\boldsymbol{V}_{\text{null}}^\top \tilde{\boldsymbol{\mu}}_0, \boldsymbol{V}_{\text{null}}^\top \tilde{\boldsymbol{\Sigma}}_0 \boldsymbol{V}_{\text{null}}\Big)\Big)
\end{aligned}$$

where we used that $\boldsymbol{P}_{\mathcal{S}} \tilde{\boldsymbol{\mu}} = \hat{\boldsymbol{\mu}}$ for any $(\tilde{\boldsymbol{\mu}}, \tilde{\boldsymbol{S}})$ in the feasible set $\Theta_\star$. Therefore it suffices to show that the projected solution $\tilde{\boldsymbol{\theta}}_\star^{\text{rGD}}$ minimizes

$$\mathrm{W}_2^2\Big(\mathcal{N}\Big(\boldsymbol{V}_{\text{null}}^\top \tilde{\boldsymbol{\mu}}, \boldsymbol{V}_{\text{null}}^\top \tilde{\boldsymbol{\Sigma}} \boldsymbol{V}_{\text{null}}\Big), \mathcal{N}\Big(\boldsymbol{V}_{\text{null}}^\top \tilde{\boldsymbol{\mu}}_0, \boldsymbol{V}_{\text{null}}^\top \tilde{\boldsymbol{\Sigma}}_0 \boldsymbol{V}_{\text{null}}\Big)\Big) \geq 0. \tag{S95}$$

We have using the definition of the iterates in Equation (9) that

$$\boldsymbol{V}_{\text{null}}^\top \tilde{\boldsymbol{\mu}}_\star^{\text{rGD}} = \boldsymbol{V}_{\text{null}}^\top \boldsymbol{P}_{\boldsymbol{V}} \left(\lim_{t\to\infty} \frac{1}{\log(t)} \boldsymbol{\mu}_t + \boldsymbol{P}_{\text{null}(\boldsymbol{X})} \boldsymbol{\mu}_0 \right) \tag{S96}$$

$$= \boldsymbol{V}_{\text{null}}^\top (\hat{\boldsymbol{\mu}} + \boldsymbol{P}_{\text{null}(\boldsymbol{X})} \boldsymbol{\mu}_0) = \boldsymbol{V}_{\text{null}}^\top \boldsymbol{\mu}_0 \tag{S97}$$

where we used $\hat{\boldsymbol{\mu}} \in \operatorname{range}(\boldsymbol{X}_{\mathcal{S}}^\top)$. Further, it holds for the gradient of the expected loss (S23) with respect to the covariance factor parameters that

$$\boldsymbol{V}_{\text{null}}^\top \tilde{\boldsymbol{S}}_\star^{\text{rGD}} = \boldsymbol{V}_{\text{null}}^\top \boldsymbol{P}_{\boldsymbol{V}} \boldsymbol{S}_\star^{\text{rGD}} = \boldsymbol{V}_{\text{null}}^\top \boldsymbol{S}_\star^{\text{rGD}} = \boldsymbol{V}_{\text{null}}^\top \Bigg(\boldsymbol{S}_0 - \underbrace{\sum_{t=1}^{\infty} \eta_t \nabla_{\boldsymbol{S}} \bar{\ell}(\boldsymbol{\mu}_t, \boldsymbol{S}_t)}_{\in \operatorname{range}(\boldsymbol{X}^\top)} \Bigg) \tag{S98}$$

$$= \boldsymbol{V}_{\text{null}}^\top \boldsymbol{S}_0 = \boldsymbol{V}_{\text{null}}^\top \boldsymbol{P}_{\boldsymbol{V}} \boldsymbol{S}_0 = \boldsymbol{V}_{\text{null}}^\top \tilde{\boldsymbol{S}}_0. \tag{S99}$$

Therefore we have that

$$\mathrm{W}_2^2\Big(\mathcal{N}\Big(\boldsymbol{V}_{\text{null}}^\top \tilde{\boldsymbol{\mu}}_\star^{\text{rGD}}, \boldsymbol{V}_{\text{null}}^\top \tilde{\boldsymbol{\Sigma}}_\star^{\text{rGD}} \boldsymbol{V}_{\text{null}}\Big), \mathcal{N}\Big(\boldsymbol{V}_{\text{null}}^\top \tilde{\boldsymbol{\mu}}_0, \boldsymbol{V}_{\text{null}}^\top \tilde{\boldsymbol{\Sigma}}_0 \boldsymbol{V}_{\text{null}}\Big)\Big) = 0 \tag{S100}$$

and thus the projected variational parameters $\tilde{\boldsymbol{\theta}}_\star^{\text{rGD}}$ are both feasible (S90) and minimize the squared 2-Wasserstein distance to the projected initialization / prior (S95). This completes the proof for the generalized version of Theorem 2 without Assumption 2, which we state here for convenience.

Lemma S6
Given the assumptions of Theorem 2, except for Assumption 2 meaning the support vectors $\boldsymbol{X}_{\mathcal{S}}$ do not *necessarily span the data, it holds for the limit point of rescaled gradient descent that*

$$\boldsymbol{\theta}_\star^{\mathrm{rGD}} \in \underset{\substack{\boldsymbol{\theta}=(\boldsymbol{\mu},\boldsymbol{S}) \\ s.t.\ \boldsymbol{\theta}\in\Theta_\star}}{\arg\min}\, \mathrm{W}_2^2\left(q_{\boldsymbol{\theta}}^{\mathrm{proj}}, p^{\mathrm{proj}}\right). \tag{S101}$$

If in addition Assumption 2 holds, i.e. the support vectors span the training data $\boldsymbol{X}$, such that

$$\operatorname{span}(\{\boldsymbol{x}_n\}_{n\in[N]}) = \operatorname{span}(\{\boldsymbol{x}_n\}_{n\in\mathcal{S}}), \tag{S102}$$

then the orthogonal complement of the support vectors in $\operatorname{range}(\boldsymbol{X}^\top)$ has dimension $N - P_{\mathcal{S}} = 0$ and thus the projection $\boldsymbol{P}_{\boldsymbol{V}} = \boldsymbol{I}_{P\times P}$ is the identity and therefore

$$q_{\boldsymbol{\theta}}^{\mathrm{proj}} = q_{\boldsymbol{\theta}} \qquad \text{and} \qquad p^{\mathrm{proj}} = p. \tag{S103}$$

This completes the proof of Theorem 2.

□

S1.3 NLL Overfitting and the Need for (Temperature) Scaling

In Theorem 2, we assume we rescale the mean parameters. This is because the exponential loss can be made arbitrarily small for a mean vector that is aligned with the L_2 max-margin vector simply by increasing its magnitude. In fact, the sequence of mean parameters identified by gradient descent diverges to infinity at a logarithmic rate $\boldsymbol{\mu}_t^{\mathrm{GD}} \approx \log(t)\hat{\boldsymbol{\mu}}$ as we show[3] in Lemma S5 and illustrate in Figure S2 (right panel).

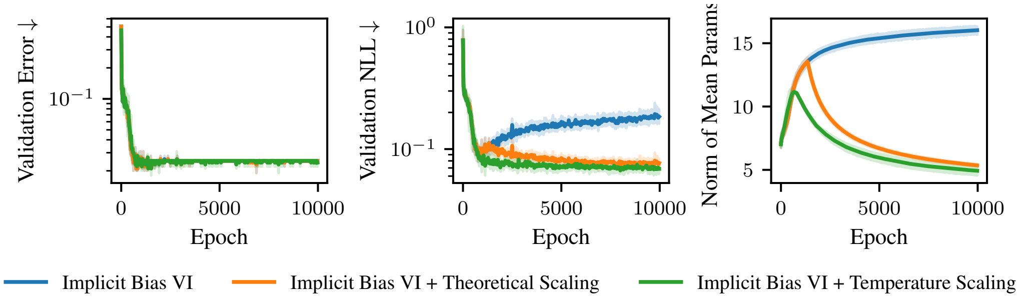


Figure S2: *NLL overfitting in classification due to implicit bias of the mean parameters.* As shown here for a two-hidden layer neural network on synthetic data, when training with vanilla SGD the mean parameters diverge to infinity $\|\boldsymbol{\mu}_t\|_2 \approx \mathcal{O}(\log(t))$ (right) and thus the classifier will eventually overfit in terms of negative log-likelihood (left and middle). Rescaling the GD iterates as in Theorem 2 or using temperature scaling [68] avoids overfitting.

This bias of the mean parameters towards the max-margin solution does not impact the train loss or validation error, but leads to overfitting in terms of validation NLL (see Figure S2) as long as there is at least one misclassified datapoint $\boldsymbol{x}$, since then the (average) validation NLL is given by

$$\begin{aligned}\bar{\ell}(\boldsymbol{\theta}_t^{\mathrm{GD}}) &= \mathbb{E}_{q_{\boldsymbol{\theta}_t^{\mathrm{GD}}}(\boldsymbol{w})}\left(\exp(-y\boldsymbol{x}^\top\boldsymbol{w})\right) = \exp(\boldsymbol{x}^\top\boldsymbol{\mu}_t^{\mathrm{GD}} + \tfrac{1}{2}\boldsymbol{x}^\top\boldsymbol{S}_t^{\mathrm{GD}}(\boldsymbol{S}_t^{\mathrm{GD}})^\top\boldsymbol{x}) \\ &\approx \exp(\log(t)\boldsymbol{x}^\top\hat{\boldsymbol{\mu}} + \tfrac{1}{2}\boldsymbol{x}^\top\boldsymbol{S}_t^{\mathrm{GD}}(\boldsymbol{S}_t^{\mathrm{GD}})^\top\boldsymbol{x}) \to \infty \quad \text{as} \quad t\to\infty.\end{aligned} \tag{S104}$$

However, by rescaling the mean parameters as we do in Theorem 2, this can be prevented as Figure S2 (middle panel) illustrates for a two-hidden layer neural network on synthetic data. Such overfitting in terms of NLL has been studied extensively empirically with the perhaps most common remedy being Temperature Scaling (TS) [68]. As we show empirically in Figure S2, instead

[3] This has been observed previously in the deterministic case (see Theorem 3 of Soudry et al. [4]) and thus naturally also appears in our probabilistic extension.

of using the theoretical rescaling, using temperature scaling performs very well, especially in the non-asymptotic regime, which is why we also adopt it for our experiments in Section 5.

The aforementioned divergence of the mean parameters to infinity also explains the need for the projection of the prior mean parameters in Equation (9), since any bias from the initialization vanishes in the limit of infinite training. At first glance the additional projection seems computationally prohibitive for anything but a zero mean prior, but close inspection of the implicit bias of the covariance parameters $\boldsymbol{S}$ in Theorem 2 shows that at convergence

$$\forall n : \operatorname{Var}_{q_{\boldsymbol{\theta}}}(f_{\boldsymbol{w}}(\boldsymbol{x}_n)) = \boldsymbol{x}_n^\top \boldsymbol{S}\boldsymbol{S}^\top \boldsymbol{x}_n = 0 \implies \operatorname{range}(\boldsymbol{S}) \subset \operatorname{null}(\boldsymbol{X}) \tag{S105}$$

Meaning we can approximate a basis of the null space of the training data by computing a QR decomposition of the covariance factor in $\mathcal{O}(PR^2)$ once at the end of training. For $R = P$ the inclusion becomes an equality and the projection can be computed exactly.

S2 Parametrization, Feature Learning and Hyperparameter Transfer

The inductive bias of SGD in the variational setting is determined by the initialization and variational parametrization, and is formalized in Theorem 1. What is not covered, but not unusual in practice, are layer-specific learning rates. Luckily, these can be absorbed into the weights of the model and the initialization, resulting in a single global learning rate [Lemma J.1, 67]. We refer to this set of choices — initialization, (variational) parameters and layer-specific learning rates — as the *parametrization* of a variational neural network in analogy to how the term is used in deep learning. While parameterization is well-studied for non-probabilistic deep learning, it has been identified as one of the "grand challenges of Bayesian computation" [80].

The "standard parameterization" (SP) initializes the weights of a neural network randomly from a distribution with variance $\propto 1/\texttt{fan\_in}$ (e.g., as in Kaiming initialization, the PyTorch default) and makes no further adjustments to the forward pass or learning rate. In contrast, the maximal update parametrization (μP) [66] ensures feature learning even as the width of the network tends to infinity. Feature learning is at the core of the modern deep learning pipeline, permitting foundation models to extract features from large datasets that are then fine-tuned. Additionally, under μP, hyperparameters like the learning rate, can be tuned on a small model and transferred to a large-scale model [67].

Given our interpretation of training via the expected loss as generalized variational inference with *a prior that is implied by the parametrization*, a natural question is whether we can extend μP to the variational setting and thus inherit its inductive bias. In the probabilistic setting, feature learning now occurs when the *distribution* over hidden units changes from initialization. At any point during training, the ith hidden unit in layer l is a function of four random variables: the variational mean and covariance parameters $(\boldsymbol{\mu}, \boldsymbol{S})$, Gaussian noise $\boldsymbol{z}$, and the previous layer hidden units:

$$\boldsymbol{h}_i^{(l)}(\boldsymbol{x}) = \boldsymbol{W}_i \boldsymbol{h}^{(l-1)}(\boldsymbol{x}) = (\boldsymbol{\mu}_i + \boldsymbol{S}_i \boldsymbol{z})\boldsymbol{h}^{(l-1)}(\boldsymbol{x}). \tag{S106}$$

The parameters are random because of the stochasticity in the initialization and/or optimization procedure, while the noise is randomly drawn during each forward pass. Since the $\boldsymbol{S}_i\boldsymbol{z}$ term is a sum over R terms, where R is the rank of $\boldsymbol{S} \in \mathbb{R}^{P \times R}$, applying the central limit theorem we propose scaling this term by $R^{-1/2}$ and then applying μP to the mean and covariance parameters. In practice, we implement the scaling via an adjustment to the covariance initialization and learning rate.

In this section we discuss the role of parameterization at initialization, feature learning in the last hidden layer as the width is increased, and how parameterization can enable hyperparameter transfer.

Notation For this section we need a more detailed neural network notation. Denote an L-hidden layer, width-D feedforward neural network by $f(\boldsymbol{x}) \in \mathbb{R}^D_{\text{out}}$, with inputs $\boldsymbol{x} \in \mathbb{R}^{D_{\text{in}}}$, weights $\boldsymbol{W}^{(l)}$, pre-activations $\boldsymbol{h}^{(l)}(\boldsymbol{x}) \in \mathbb{R}^{D^{(l)}}$, and post-activations (or "features") $\boldsymbol{g}^{(l)}(\boldsymbol{x}) \in \mathbb{R}^{D^{(l)}}$. That is, $\boldsymbol{h}^{(1)}(\boldsymbol{x}) = \boldsymbol{W}^{(1)}\boldsymbol{x}$ and, for $l \in 1, \dots, L-1$,

$$\boldsymbol{g}^{(l)}(\boldsymbol{x}) = \phi\left(\boldsymbol{h}^{(l)}(\boldsymbol{x})\right),\ \boldsymbol{h}^{(l+1)}(\boldsymbol{x}) = \boldsymbol{W}^{(l+1)}\boldsymbol{g}^{(l)}(\boldsymbol{x}),$$

and the network output is given by $f(\boldsymbol{x}) = \boldsymbol{W}^{(L+1)}\boldsymbol{g}^{(L)}(\boldsymbol{x})$, where $\phi(\bullet)$ is an activation function.

For convenience, we may abuse notation and write $\boldsymbol{h}^{(0)}(\boldsymbol{x}) = \boldsymbol{x}$ and $\boldsymbol{h}^{(L+1)}(\boldsymbol{x}) = f(\boldsymbol{x})$. Throughout we use $\bullet^{(l)}$ to indicate the layer, subscript $\bullet_t$ to indicate the training time (i.e., epoch),

$\Delta\bullet_t = \bullet_t - \bullet_0$ to indicate the change since initialization, and $[\bullet]_i$, $[\bullet]_{ij}$ to indicate the component within a vector or matrix.

S2.1 Definitions of Stability and Feature Learning

The following definitions extend those of Yang and Hu [66] to the variational setting.

Definition S1 (*bc* scaling)
In layer l, the variational parameters are initialized as

$$[\boldsymbol{\mu}_0^{(l)}]_i \sim \mathcal{N}\Big(0, D^{-2b^{(l)}}\Big), \quad [\boldsymbol{S}_0^{(l)}]_{ij} \sim \mathcal{N}\Big(0, D^{-2\tilde{b}^{(l)}}\Big)$$

and the learning rates for the mean and covariance parameters, respectively, are set to

$$\eta^{(l)} = \eta D^{-c^{(l)}}, \tilde{\eta}^{(l)} = \eta D^{-\tilde{c}^{(l)}}.$$

The hyperparameter η represents a global learning rate that can be tuned, as for example in the hyperparameter transfer experiment from Section S2.4.

For the next two definitions, let $m_r(X) = \mathbb{E}_{\boldsymbol{z}}((X - \mathbb{E}_{\boldsymbol{z}}(X))^r)$ denote the rth central moment moment of a random variable X with respect to $\boldsymbol{z}$, which represents all reparameterization noise in the random variable X. All Landau notation in Section S2 refers to asymptotic behavior in width D in probability over reparameterization noise $\boldsymbol{z}$. We say that a vector sequence $\{\boldsymbol{v}_D\}_{D=1}^{\infty}$, where each $\boldsymbol{v}_D \in \mathbb{R}^D$, is $\mathcal{O}(D^{-a})$ if the scalar sequence $\{\sqrt{\frac{1}{D}\|\boldsymbol{v}_D\|^2}\}_{D=1}^{\infty} = \{\text{RMSE}(\boldsymbol{v}_D)\}_{D=1}^{\infty}$ is $\mathcal{O}(D^{-a})$.

Definition S2 (Stability of Moment r)
A neural network is *stable in moment* r, if all of the following hold for all $\boldsymbol{x}$ and $l \in \{1, \ldots, L\}$.

1. At initialization ($t = 0$):
   (a) The pre- and post-activations are $\Theta(1)$:
   $$m_r(\boldsymbol{h}_0^{(l)}(\boldsymbol{x})), m_r(\boldsymbol{g}_0^{(l)}(\boldsymbol{x})) = \Theta(1)$$
   (b) The function is $\mathcal{O}(1)$:
   $$m_r(f_0(\boldsymbol{x})) = \mathcal{O}(1)$$
2. At any point during training $t > 0$:
   (a) The change from initialization in the pre- and post-activations are $\mathcal{O}(1)$:
   $$\Delta m_r(\boldsymbol{h}_t^{(l)}(\boldsymbol{x})), \Delta m_r(\boldsymbol{g}_t^{(l)}(\boldsymbol{x})) = \mathcal{O}(1)$$
   (b) The function is $\mathcal{O}(1)$:
   $$m_r(f_t(\boldsymbol{x})) = \mathcal{O}(1)$$

Definition S3 (Feature Learning of Moment r)
Feature learning occurs in moment r in layer l if, for any $t > 0$, the change from initialization is $\Omega(1)$:

$$\Delta m_r\left(\boldsymbol{g}_t^{(l)}(\boldsymbol{x})\right) = \Omega(1).$$

As we will see later, Figure S5 and Figure S6 investigate feature learning for the first two moments.

S2.2 Initialization Scaling for a Linear Network

In this section we illustrate how the initialization scaling $\{(b^{(l)}, \tilde{b}^{(l)})\}$ can be chosen for stability. For simplicity, we consider a linear feedforward network of width D evaluated on a single input $\boldsymbol{x} \in \mathbb{R}^D_{\text{in}}$. We assume a Gaussian variational family that factorizes across layers. This implies the hidden units evolve as $\boldsymbol{h}_t^{(l+1)} = \boldsymbol{W}_t^{(l+1)}\boldsymbol{h}_t^{(l)}$ and the weights are linked to the variational parameters by $\text{vec}(\boldsymbol{W}_t^{(l)}) = \boldsymbol{\mu}_t^{(l)} + \boldsymbol{S}_t^{(l)}\boldsymbol{z}$.

Therefore, the mean and variance of the ith component hidden units in layer $l \in \{1, \dots, L+1\}$, where $i \in 1 \dots, D^{(l)}$, are given by

$$\begin{aligned}
\mathbb{E}_z\Big([\boldsymbol{h}_t^{(l)}]_i\Big) &= [\boldsymbol{\mu}_t^{(l)}]_I^\mathsf{T}\, \mathbb{E}_z\Big(\boldsymbol{h}_t^{(l-1)}\Big) \\
\mathrm{Var}_z\Big([\boldsymbol{h}_t^{(l)}]_i\Big) &= [\boldsymbol{\mu}_t^{(l)}]_I^\mathsf{T} \boldsymbol{C}_t^{(l-1)} [\boldsymbol{\mu}^{(l)}]_I + \mathrm{tr}([\boldsymbol{S}_t^{(l)}]_{I,:}^\mathsf{T} \boldsymbol{A}_t^{(l-1)} [\boldsymbol{S}_t^{(l)}]_{I,:}),
\end{aligned}$$

where $I = \{iD^{(l-1)}, \dots, (i{+}1)D^{(l-1)}\}$ and the second moment of and covariance of layer-l hidden units are denoted by

$$\begin{aligned}
\boldsymbol{A}_t^{(l)} &= \mathbb{E}_z\Big(\boldsymbol{h}_t^{(l)} \boldsymbol{h}_t^{((l))\mathsf{T}}\Big) \\
\boldsymbol{C}_t^{(l)} &= \boldsymbol{A}_t^{(l)} - \mathbb{E}_z\Big(\boldsymbol{h}_t^{(l)}\Big)\, \mathbb{E}_z\Big(\boldsymbol{h}_t^{(l)}\Big)^\mathsf{T}.
\end{aligned}$$

Mean We start with the mean of the hidden units, which conveniently depends only on the mean variational parameters and the previous layer hidden units.

$$\begin{aligned}
\mathbb{E}_z\Big([\boldsymbol{h}_0^{(l)}]_i\Big) &= \sum_{j=1}^{D^{(l-1)}} [\boldsymbol{\mu}_0^{(l)}]_{I_j}\, \mathbb{E}_z\Big([\boldsymbol{h}_0^{(l-1)}]_j\Big) \\
&= \mathcal{O}\Big(\sqrt{D^{(l-1)}} \cdot D^{-b^{(l)}} \cdot 1\Big) \\
&= \begin{cases} \mathcal{O}\Big(D^{-b^{(1)}}\Big) & l = 1 \\ \mathcal{O}\Big(D^{-(b^{(l)} - \frac{1}{2})}\Big) & l \in \{2, \dots, L+1\} \end{cases}.
\end{aligned}$$

Therefore, we require $b^{(1)} \geq 0$ and $b^{(l)} \geq \frac{1}{2}$ for $l \in \{2, \dots, L+1\}$.

Variance Next we examine the variance of hidden units. Consider the first term, which represents the contribution of the mean parameters.

$$\begin{aligned}
[\boldsymbol{\mu}_0^{(l)}]_I^\mathsf{T} \boldsymbol{C}_0^{(l-1)} [\boldsymbol{\mu}^{(l)}]_I &= \sum_{j=1}^{D^{(l-1)}} [\boldsymbol{\mu}_0^{(l)}]_{I_j}^2 [\boldsymbol{C}_0^{(l-1)}]_{j,j} + \sum_{j \neq j'}^{D^{(l-1)}} [\boldsymbol{\mu}_0^{(l)}]_{I_j} [\boldsymbol{C}_0^{(l-1)}]_{j,j'} [\boldsymbol{\mu}_0^{(l)}]_{I_{j'}} \\
&= \mathcal{O}\Big(D^{(l-1)} \cdot D^{-2b^{(l)}} \cdot 1\Big) + \mathcal{O}\Bigg(\sqrt{D^{(l-1)}(D^{(l-1)} - 1)} \cdot D^{-b^{(l)}} \cdot 1 \cdot D^{-b^{(l)}}\Bigg) \\
&= \mathcal{O}\Big(D^{(l-1)} \cdot D^{-2b^{(l)}}\Big) \\
&= \begin{cases} \mathcal{O}\Big(D^{-2b^{(1)}}\Big) & l = 1 \\ \mathcal{O}\Big(D^{-(2b^{(l)} - 1)}\Big) & l \in l \in \{2, \dots, L+1\}. \end{cases}
\end{aligned}$$

Therefore, we require $b^{(1)} \geq 0$ and $b^{(l)} \geq \frac{1}{2}$ for $l \in \{2, \dots, L+1\}$. Notice these are the same requirements as above for the mean of the hidden units. We summarize the scaling for the mean parameters as

$$\boxed{b^{(l)} \geq \begin{cases} 0 & l = 1 \\ \frac{1}{2} & l \in \{2, \dots, L+1\}. \end{cases}} \tag{S107}$$

Now consider the second term in the variance of the hidden units. Assume the rank scales with the input and output dimension of a layer as $R^{(l)} = (D^{(l-1)} D^{(l)})^{p^{(l)}}$, where $p^{(l)} \in [0, 1]$.

$$\begin{aligned}
\mathrm{tr}([\boldsymbol{S}_0^{(l)}]_{I,:}^\mathsf{T} \boldsymbol{A}_0^{(l-1)} [\boldsymbol{S}_0^{(l)}]_{I,:}) &= \sum_{r=1}^{R^{(l)}} [\boldsymbol{S}_0^{(l)}]_{I,r}^\mathsf{T} \boldsymbol{A}_0^{(l-1)} [\boldsymbol{S}_0^{(l)}]_{I,r} \\
&= \sum_{r=1}^{R^{(l)}} \left(\sum_{j=1}^{D^{(l-1)}} [\boldsymbol{S}_0^{(l)}]_{I_j,r}^2 [\boldsymbol{A}_0^{(l-1)}]_{j,j} + \sum_{j \neq j'}^{D^{(l-1)}} [\boldsymbol{S}_0^{(l)}]_{I_j,r} [\boldsymbol{A}_0^{(l-1)}]_{j,j'} [\boldsymbol{S}_0^{(l)}]_{I_{j'},r} \right)
\end{aligned}$$

$$= \mathcal{O}\Big(R^{(l)} D^{(l-1)} \cdot D^{-2\tilde{b}^{(l)}} \cdot 1\Big) + \mathcal{O}\Bigg(\sqrt{R^{(l)} D^{(l-1)} (D^{(l-1)} - 1)} \cdot D^{-\tilde{b}^{(l)}} \cdot 1 \cdot D^{-\tilde{b}^{(l)}}\Bigg)$$

$$= \mathcal{O}\Big(R^{(l)} D^{(l-1)} D^{-2\tilde{b}^{(l)}}\Big)$$

$$= \begin{cases} \mathcal{O}\Big(D^{-(2\tilde{b}^{(1)} - p^{(1)})}\Big) & l = 1 \\ \mathcal{O}\Big(D^{-(2\tilde{b}^{(l)} - 1 - 2p^{(l)})}\Big) & l \in \{2, \dots, L\} \\ \mathcal{O}\Big(D^{-(2\tilde{b}^{(L+1)} - 1 - p^{(L+1)})}\Big) & l = L + 1. \end{cases}$$

Therefore we require $\tilde{b}^{(0)} \geq \frac{p^{(1)}}{2}$, $\tilde{b}^{(l)} \geq \frac{1}{2} + p^{(l)}$ for $l \in \{2, \dots, L\}$, and $\tilde{b}^{(L+1)} \geq \frac{1}{2} + \frac{p^{(L+1)}}{2}$. Notice we can write these conditions in terms of the mean scaling as

$$\boxed{\tilde{b}^{(l)} \geq b^{(l)} + \begin{cases} \frac{p^{(l)}}{2} & l = 1 \\ p^{(l)} & l \in \{2, \dots, L\} \\ \frac{p^{(l)}}{2} & l = L + 1. \end{cases}} \tag{S108}$$

S2.3 Proposed Scaling

The previous section derives the necessary conditions for stability at initialization. Recall we propose scaling the contribution of the covariance parameters to the forward pass, i.e. the $\boldsymbol{S}\boldsymbol{z}$ term, by $R^{-1/2}$ since each element in the term is a sum over R random variables, where R is the rank of $\boldsymbol{S}$. In the more detailed notation of this section, the proposed scaling implies the forward pass in a linear layer is given by

$$[\boldsymbol{h}_t^{(l)}]_i = [\boldsymbol{W}_t]_{:,i} \boldsymbol{h}_t^{(l-1)} = \Big([\boldsymbol{\mu}_t^{(l)}]_I + R^{-1/2} [\boldsymbol{S}_t^{(l)}]_I \boldsymbol{z}^{(l)}\Big) \boldsymbol{h}_t^{(l-1)}. \tag{S109}$$

In practice, rather than scaling $[\boldsymbol{S}_t^{(l)}]_I \boldsymbol{z}^{(l)}$ by $R^{-1/2}$ in the forward pass, we apply Lemma J.1 from Yang et al. [67] to instead scale the initialization by $R^{-1/2}$ and, in SGD, the learning rate by R^{-1}. Scaling by the rank allows treating the mean and covariance parameters as if they were weights parameterized by μP in a non-probabilistic network, inheriting any scaling that has already been derived for that architecture.

From Table 3 of Yang et al. [67], we therefore scale the mean parameters as

$$b^{(l)} = \begin{cases} 0 & l = 1 \\ 1/2 & l \in \{2, \dots, L\} \\ 1 & l = L + 1 \end{cases} \quad \text{and} \quad c^{(l)} = \begin{cases} -1 & l = 1 \\ 0 & l \in \{2, \dots, L\} \\ 1 & l = L + 1. \end{cases} \tag{S110}$$

Assuming $R^{(l)} = (D^{(l-1)} D^{(l)})^{p^{(l)}}$ as before, where $p^{(l)} \in [0, 1]$, we the scale the covariance parameters as

$$\tilde{b}^{(l)} = b^{(l)} + \begin{cases} \frac{p^{(l)}}{2} & l = 1 \\ p^{(l)} & l \in \{2, \dots, L\} \\ \frac{p^{(l)}}{2} & l = L + 1 \end{cases} \quad \text{and} \quad \tilde{c}^{(l)} = c^{(l)} + \begin{cases} p^{(l)} & l = 1 \\ 2p^{(l)} & l \in \{2, \dots, L\} \\ p^{(l)} & l = L + 1. \end{cases} \tag{S111}$$

By comparing to Equations S107 and S108, we see the mean and covariance parameters in all but the output layer are initialized as large as possible while still maintaining stability. The output layer parameters scale to zero faster, since, as in μP for the weights of non-probabilistic networks, we set $b^{(L+1)}$ to 1 instead of $1/2$.

Note that in Section S2.2 we did not consider input and output dimensions that scaled with the width D for simplicity. For our experiments, we take the exact μP initialization and learning rate scaling from Yang et al. [67] — which includes, for example, a $1/$`fan_in` scaling in the input layer — for the means and then make the rank adjustment for the covariance parameters as described above.

We investigate the proposed scaling in Figures S4 and S5. We train two-hidden-layer ($L = 2$) MLPs of hidden sizes 8, 16, 32, and 64 on a single observation $(x, y) = (1, 1)$ using a squared error loss.

We use SGD with a learning rate of 0.05. For the variational networks, we assume a multivariate Gaussian variational family with a full rank covariance.

Figures S3 and S4 show the RMSE of the change in the hidden units from initialization, $\Delta \boldsymbol{g}_t^{(l)}(x) = \boldsymbol{g}_t^{(l)}(x) - \boldsymbol{g}_0^{(l)}(x)$, as a function of the hidden size. The RMSE of the hidden units *at* initialization, $\boldsymbol{g}_0^{(l)}$ is also shown in blue. Each panel corresponds to a layer of the network, so the first two panels correspond to features $\boldsymbol{g}_t^{(1)}(x)$ and $\boldsymbol{g}_t^{(2)}(x)$, respectively, while the third panel corresponds to the output of the network, $\boldsymbol{g}_t^{(3)}(x) = f_t(x)$. The difference between the figures is the paramaterization. Figure S3 uses standard parameterization (SP) while Figure S4 uses maximal update parametrization (μP). We observe that (a) the features change more under μP than SP and (b) training is more stable across hidden sizes under μP than SP, especially for smaller networks.

Figures S5 and S6 show the analogous results for a variational network. The top row shows the change in the mean of the hidden units, while the bottom row shows the change in the standard deviation. As in the non-probabilistic case, we observe that (a) both the mean and standard deviation of the features change more under μP than SP and (b) training is more stable across hidden sizes under μP than SP, especially for smaller networks.

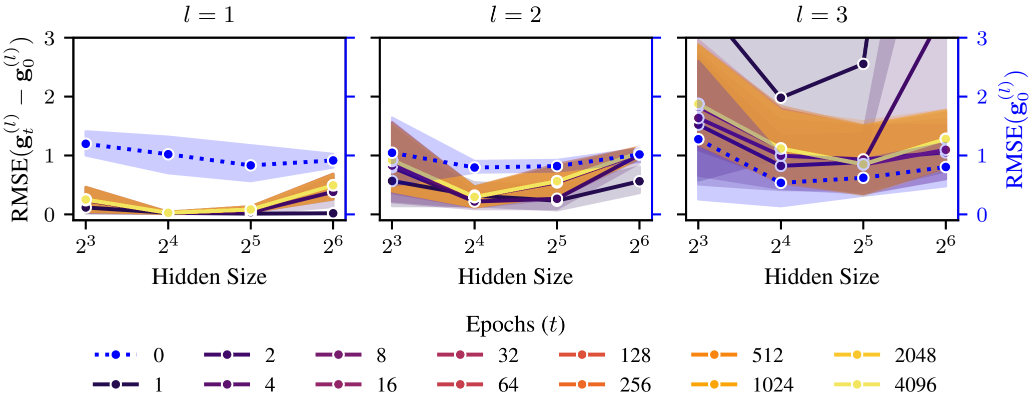


Figure S3: *MLP, Standard Parameterization.* RMSE of the change in the hidden units and, in blue, their initial values. Shaded region represents 95% confidence interval over 5 random initializations. The MLP is trained under SP.

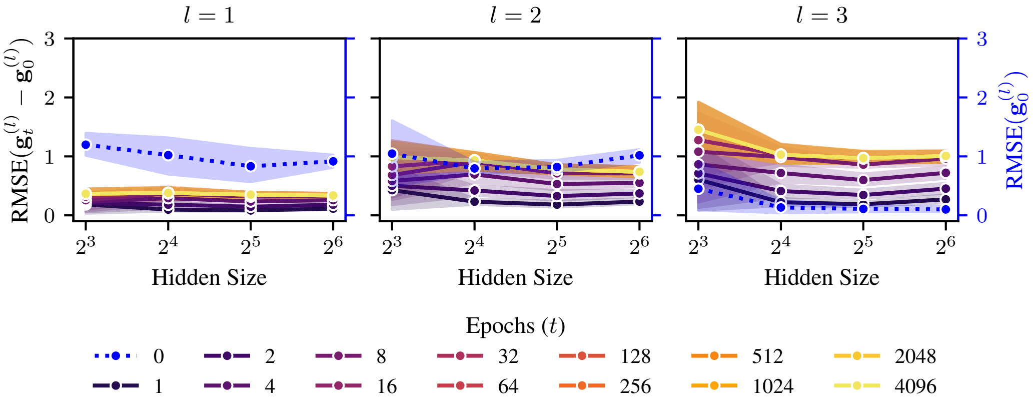


Figure S4: *MLP, Maximal Update Parameterization.* RMSE of the change in the hidden units and, in blue, their initial values. Shaded region represents 95% confidence interval over 5 random initializations. The MLP is trained under μP.

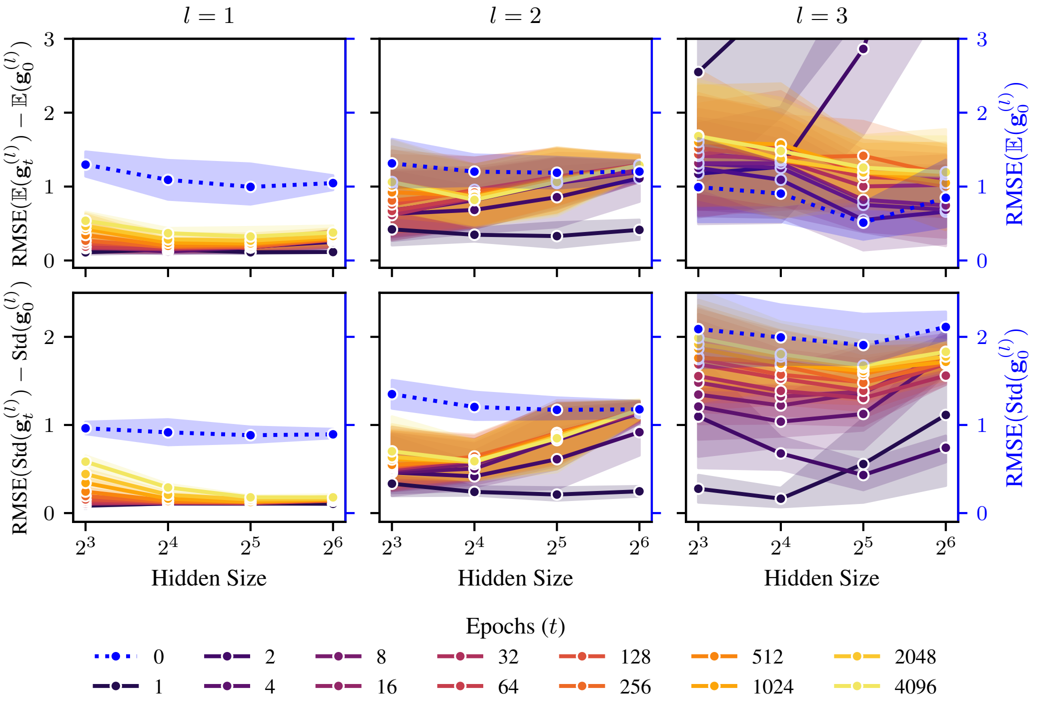


Figure S5: *Variational MLP, Standard Parameterization.* RMSE of the change in the hidden units and, in blue, their initial values. Shaded region represents 95% confidence interval over 5 random initializations. The variational MLP is trained under SP with a full rank covariance in each layer.

S2.4 Hyperparameter Transfer Experiment

Figure S7 demonstrates that our proposed maximal update parametrization enables hyperparameter transfer in a probabilistic model. We train two-hidden-layer MLPs on CIFAR10, using a low rank covariance in the final two layers. Under standard parametrization (left panel), the learning rate that results in the smallest training loss decreases with hidden size. In contrast, under μP (middle panel), it remains the same across hidden sizes. The right panel of Fig. S7 demonstrates the practical implications for model selection. For each parametrization and each hidden size D, we select the learning rate based on a grid search. In "transferred grid search" we do a grid search using the smallest model (hidden size 128) and transfer the best validating learning rate to the hidden size D model, whereas in "grid search" we perform the grid search on the hidden size D model. Relative to the test accuracy of the best performing model across learning rate and parametrization, we see that (a) μP outperforms SP, though the gap decreases with hidden size, and (b) the transfer strategy works well for μP but poorly for SP once the hidden size exceeds 256.

The μP parametrization ensures stability and feature learning. Since we interpret the initialization as a prior, which we emphasize is fully theoretically justified in the case of a linear model, this suggests a new approach to designing priors over neural networks. Instead of eliciting beliefs about the relative likelihood of weights or functions, consider how the optimization process evolves the initial parameters and whether desirable properties, like feature learning, will be preserved.

Details on Hyperparameter Transfer Experiment We train two-hidden-layer MLPs of width 128, 256, 512, 1024, and 2048 on CIFAR-10. For comparability to Figure 3 in Tensor Programs V [67] we use the same hyperparameters but applied to the mean parameters.[4] For the input layer, we scale the mean parameters at initialization by a factor of 16 and in the forward pass by a factor of 1/16. For the output layer, we scale the mean parameters by 0.0 at initialization and by 32.0 in

[4] Specifically, we used the hyperparameters as indicated here: https://github.com/microsoft/mup/blob/main/examples/MLP/demo.ipynb

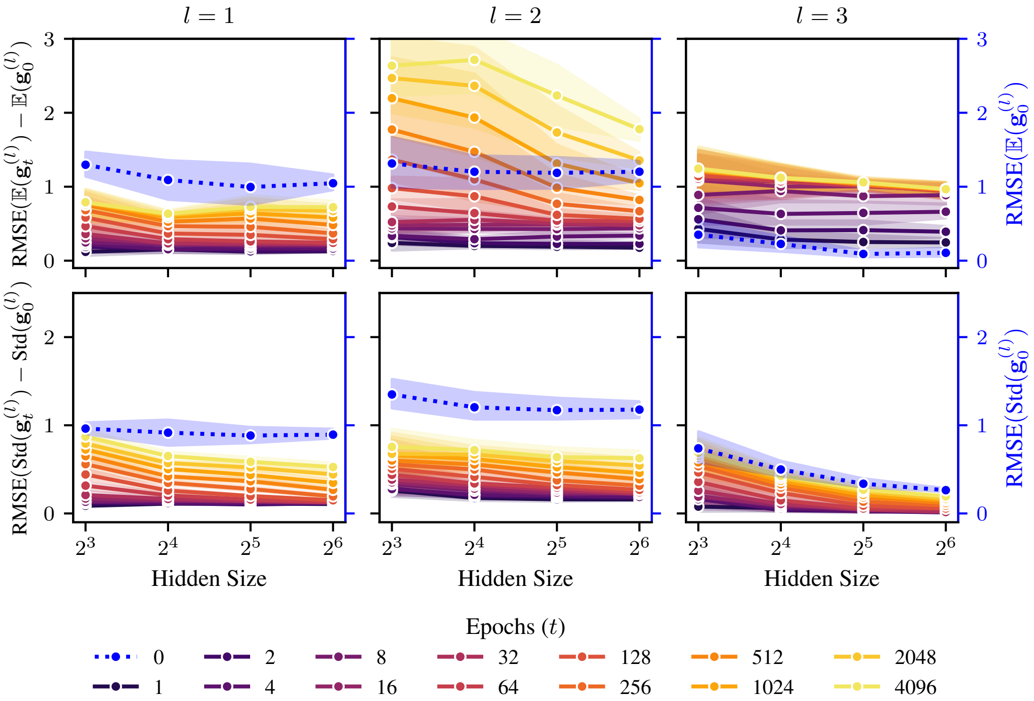


Figure S6: *Variational MLP, Maximal Update Parametrization.* RMSE of the change in the hidden units and, in blue, their initial values. Shaded region represents 95% confidence interval over 5 random initializations. The variational MLP is trained under μP with a full rank covariance in each layer.

the forward pass. We use 20 epochs, batch size 64, and a grid of global learning rates ranging from 2^{-8} to 2^0 with cosine annealing during training. For the grid search results shown in the right panel of Figure S7, we use validation NLL for model selection and then evaluate the relative test error compared to the best performing model for that width across parameterizations and learning rates.

S3 Experiments

This section outlines in more detail the experimental setup, including datasets (Section S3.1.1), metrics (Section S3.1.2), architectures, the training setup and method details (Section S3.3.1). It also contains additional experiments to the ones in the main paper (Sections S3.2, S3.3.2 and S3.3.3).

S3.1 Setup and Details

In all of our experiments we used the following datasets and metrics.

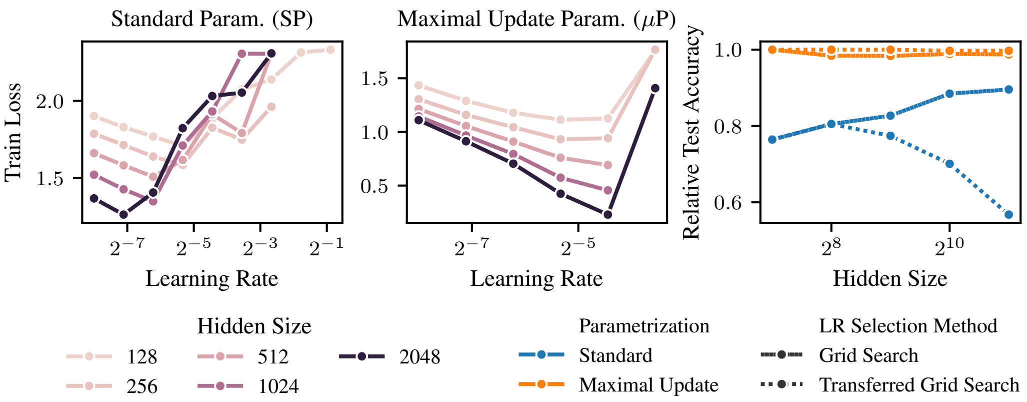


Figure S7: *Hyperparameter Transfer.* When scaling the size of a neural network, one has to re-tune the hyperparameters, such as the learning rate, when using the standard parametrization (SP). The same is true for probabilistic networks as we show here on CIFAR-10 (left). However, when using our proposed extension of the maximal update parametrization (μP) [67] to probabilistic networks, one can tune the learning rate on a small model and achieve optimal generalization for larger models by "transferring" the optimal learning rate from a smaller model (center and right).

S3.1.1 Datasets

Table S1: *Benchmark datasets used in our experiments.* All corrupted datasets are only intended for evaluation and thus only have test sets consisting of 15 different corruptions of the original test set.

Dataset	N	N_{test}	D_{in}	C	Train / Validation Split
MNIST [70]	60 000	10 000	28×28	10	(0.9, 0.1)
CIFAR-10 [81]	50 000	10 000	$3 \times 32 \times 32$	10	(0.9, 0.1)
CIFAR-100 [81]	50 000	10 000	$3 \times 32 \times 32$	100	(0.9, 0.1)
TinyImageNet [82]	100 000	10 000	$3 \times 64 \times 64$	200	(0.9, 0.1)
MNIST-C [72]	-	150 000	28×28	10	-
CIFAR-10-C [73]	-	150 000	$3 \times 32 \times 32$	10	-
CIFAR-100-C [73]	-	150 000	$3 \times 32 \times 32$	100	-
TinyImageNet-C [73]	-	150 000	$3 \times 64 \times 64$	200	-

S3.1.2 Metrics

Accuracy The (top-k) accuracy is defined as

$$\text{Accuracy}_k(\boldsymbol{y}, \hat{\boldsymbol{y}}) = \frac{1}{N_{\text{test}}} \sum_{n=1}^{N_{\text{test}}} 1_{(y_n \in \hat{y}_n^{1:k})}. \tag{S112}$$

Negative Log-Likelihood (NLL) The (normalized) negative log likelihood for classification is given by

$$\text{NLL}(\boldsymbol{y}, \hat{\boldsymbol{y}}) = -\frac{1}{N_{\text{test}}} \sum_{n=1}^{N_{\text{test}}} \log \hat{\boldsymbol{p}}_{\hat{y}_n}, \tag{S113}$$

where $\hat{\boldsymbol{p}}_{\hat{y}_n}$ is the probability a model assigns to the predicted class $\hat{y}_n$.

Expected Calibration Error (ECE) The expected calibration error measures how well a model is calibrated, i.e. how closely the predicted class probability matches the accuracy of the model. Assume the predicted probabilities of the model on the test set are binned into a given binning of the unit interval. Compute the accuracy a_j and average predicted probability $\hat{p}_j$ of each bin, then the expected calibration error is given by

$$\mathrm{ECE} = \sum_{j=1}^{J} b_j |a_j - \hat{p}_j|, \tag{S114}$$

where b_j is the fraction of datapoints in bin $j \in \{1, \dots, J\}$.

S3.2 Time and Memory-Efficient Training

To keep the time and memory overhead low during training, we would like to draw as few samples of the parameters as possible to evaluate the training objective $\bar{\ell}(\boldsymbol{\theta})$. Drawing M parameter samples for the loss increases the time and memory overhead of a forward and backward pass M times (disregarding parallelism). Therefore it is paramount for efficiency to use as few parameter samples as possible, ideally $M = 1$.

When drawing fewer samples from the variational distribution, the variance in the training loss and gradients increases. In practice this means one has to potentially choose a smaller learning rate to still achieve good performance. This is analogous to the previously observed linear relationship $N_b \propto \eta$ between the optimal batch size N_b and learning rate η [e.g., 28–30]. Figure S8 shows this relationship between the number of parameter samples used for training and the learning rate on MNIST for a two-hidden layer MLP of width 128.

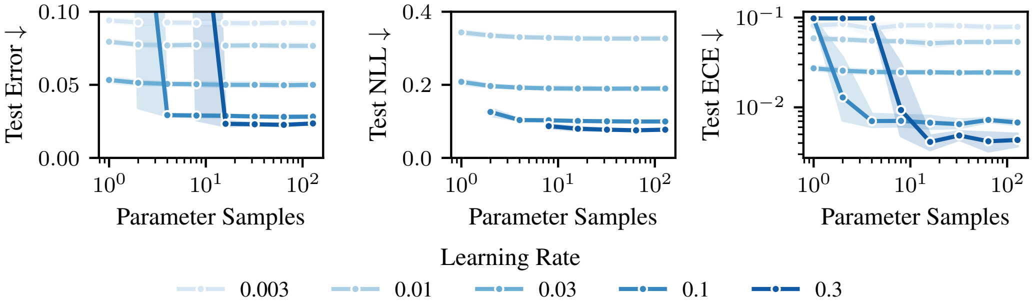


Figure S8: *Generalization versus number of parameter samples.* For a fixed number of epochs and batch size, fewer samples require a smaller learning rate. For a fixed learning rate, generalization performance quickly plateaus with more parameter samples.

As Figure S9 shows, when using momentum, generalization performance tends to increase, but only if either the number of samples is increased, or the learning rate is decreased accordingly. A similar relationship between noise in the objective and the use of momentum has previously been observed by Smith and Le [29], which propose and empirically verify a scaling law for the optimal batch size $N_b \propto \frac{\eta}{1-\gamma}$ as a function of the momentum parameter $\gamma > 0$.

S3.3 In- and Out-of-distribution Generalization

This section recounts details of the methods we benchmark in Section 5, how they are trained and additional experimental results.

S3.3.1 Architectures, Training, and Methods

Architectures We use convolutional architectures for all experiments in Section 5. For MNIST, we use a standard LeNet-5 [70] with ReLU activations. For CIFAR-10, CIFAR-100 and TinyImageNet we use a ResNet-34 [71] where the first layer is a 2D convolution with `kernel_size=3`, `stride=1` and `padding=1` to account for the image resolution of CIFAR and TinyImageNet and the normalization layers are `GroupNorm` layers. We use pretrained weights from ImageNet for all but the first and last layer of the ResNets from `torchvision` [83] and fully finetune all parameters during training.

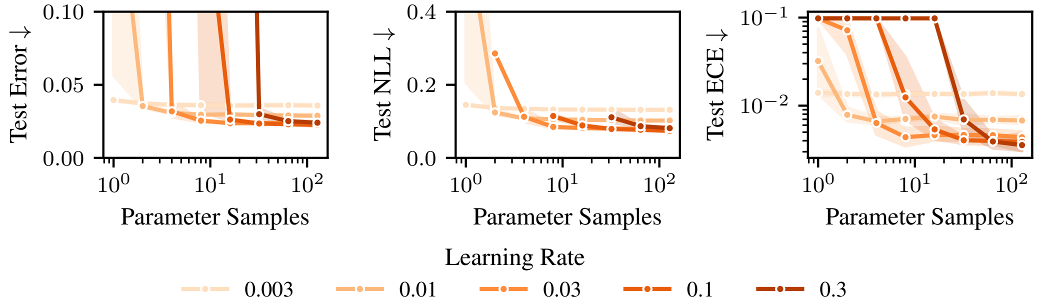


Figure S9: *Generalization versus number of parameter samples when using momentum.* Using momentum improves generalization performance, but when using fewer parameter samples, a smaller learning rate is necessary than for vanilla SGD as predicted by Equation (6).

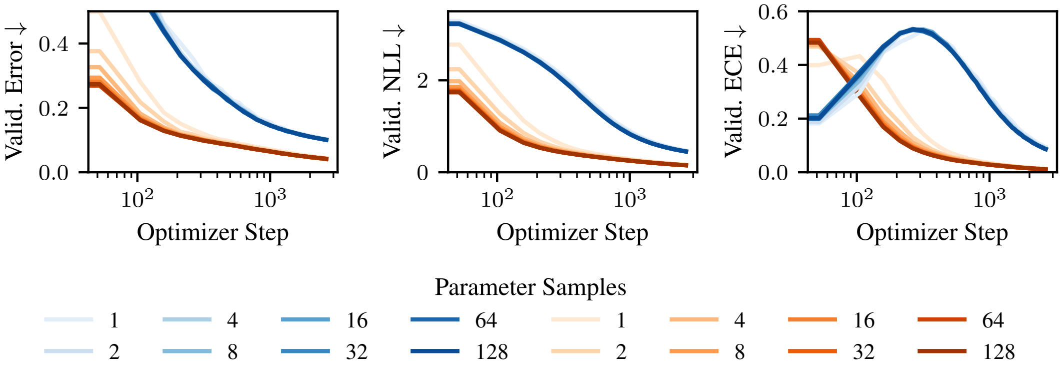


Figure S10: *Validation error during training for different numbers of parameter samples.* The difference in generalization error between different number of parameter samples vanishes with more optimization steps both for SGD (—) and when using momentum (—), *if* the learning rate is sufficiently small (in this example $\eta = 0.003$).

Training We train all models using SGD with momentum ($\gamma = 0.9$) with batch size $N_b = 128$ and learning rate $\eta = 0.005$ for 200 epochs. We do not use a learning rate scheduler since we found that neither cosine annealing nor learning rate warm-up improved the results.

Temperature Scaling [68] For temperature scaling we optimize the scalar temperature parameter in the last layer on the validation set via the L-BFGS implementation in `torch` with an initial learning rate $\eta_{\text{TS}} = 0.1$, a maximum number of 100 iterations per optimization step and `history_size=100`.

Laplace Approximation (Last-Layer, GS + ML) [49] As recommended by Daxberger et al. [49] we use a post-hoc KFAC last-layer Laplace approximation with a GGN approximation to the Hessian. We tune the hyperparameters post-hoc using type-II maximum likelihood (ML). As an alternative we also do a grid search (GS) for the prior scale, which we found to be somewhat more robust in our experiments. Finally, we compute the predictive using an (extended) probit approximation. Our implementation of the Laplace approximation is a thin wrapper of `laplace` [49] and we use its default hyperparameters throughout.

Weight-space VI (Mean-field) [34, 35] For variational inference, we used a mean-field variational family and trained via an ELBO objective with a weighting of the Kullback-Leibler regularization term to the prior. We chose a unit-variance Gaussian prior with mean that was set to the pretrained weights, except for the in- and output layer which had zero mean. We found that using a KL weight and more than a single sample (here $M = 8$) was necessary to achieve competitive performance. The KL weight was chosen to be inversely proportional to the number of parameters of the model, for which we observed better performance than a KL weight that was independent of the architecture. At test time we compute the predictive by averaging logits using 32 samples.

Implicit Bias VI [ours] For all architectures in Section 5 we use a Gaussian in- and output layer with a low-rank covariance ($R = 10, 20$). We train with a single parameter sample $M = 1$ throughout and do temperature scaling at the end of training on the validation set with the same settings as when just performing temperature scaling. We do temperature scaling in classification due to the specific form of the implicit bias in classification as described in Section S1.3. Since IBVI trains by optimizing a minibatch approximation of the expected negative log-likelihood (an average over log-probabilities with respect to parameter samples), we also average log-probabilities at test-time to compute the predictive distribution over class probabilities. Although we did not see a significant difference between averaging log-probabilities, probabilities or logits. Like for WSVI we use 32 samples at test time.

SWAG [69] We used a slightly modified implementation of SWAG based on `torch-uncertainty` and the original implementation by Maddox et al. [69]. The beginning of the averaging cycle set to half the number of total epochs and a cycle length of one, i.e. SWAG updates happen every epoch. For all other hyperparameters we use the default settings.

Deep Ensembles [52] We use five ensemble members initialized and trained independently. We compute the predictive by averaging the predicted probabilities of the ensemble members in line with standard practice [52]. We did not see a significant difference in performance between averaging logits or averaging class probabilities.

S3.3.2 In-Distribution Generalization and Uncertainty Quantification

The full results from the in-distribution generalization experiment in Section 5 can be found in Figure S11. The same experiment but done in the Maximal Update parametrization is depicted in Figure S12. When finetuning a pretrained model, we found that on some datasets (CIFAR-100, TinyImageNet) μP resulted in somewhat lower performance, contrary to the results in Section S2, where we trained from scratch. This suggests that, when pretraining, there may be a modification to the parametrization that could improve generalization.

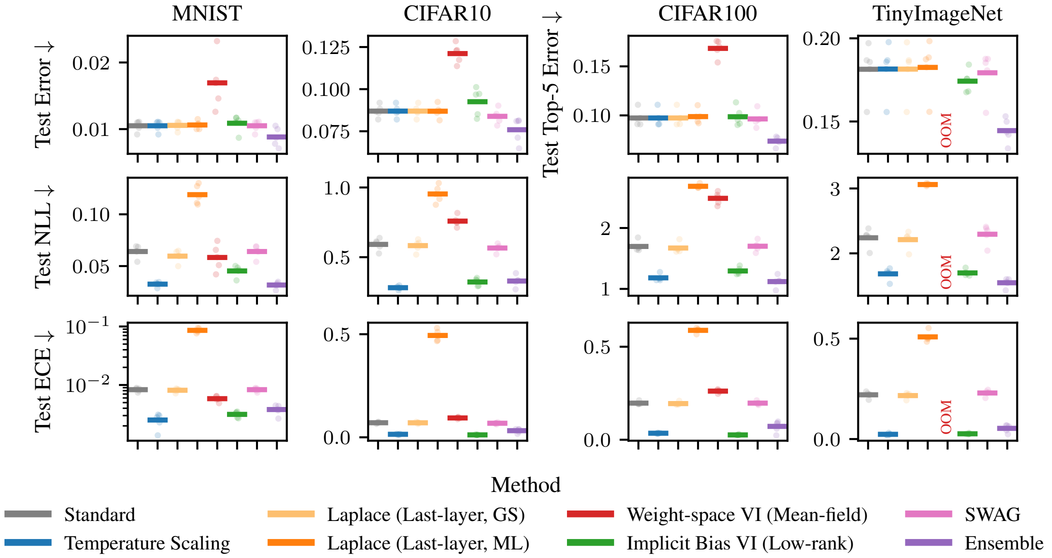


Figure S11: *In-distribution generalization and uncertainty quantification (Standard parametrization).*

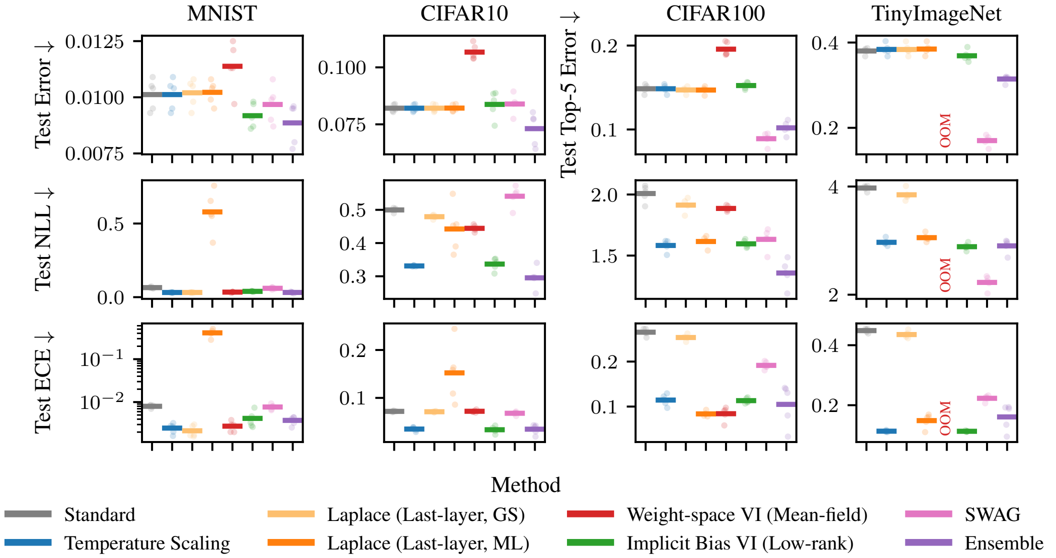


Figure S12: *In-distribution generalization and uncertainty quantification (Maximal Update parametrization).*

1148 S3.3.3 Robustness to Input Corruptions

1149 Besides the benchmark in Figure S12, we also evaluated the models trained using the Maximal
1150 Update parametrization on the corrupted datasets. The results can be found in Figure S13.

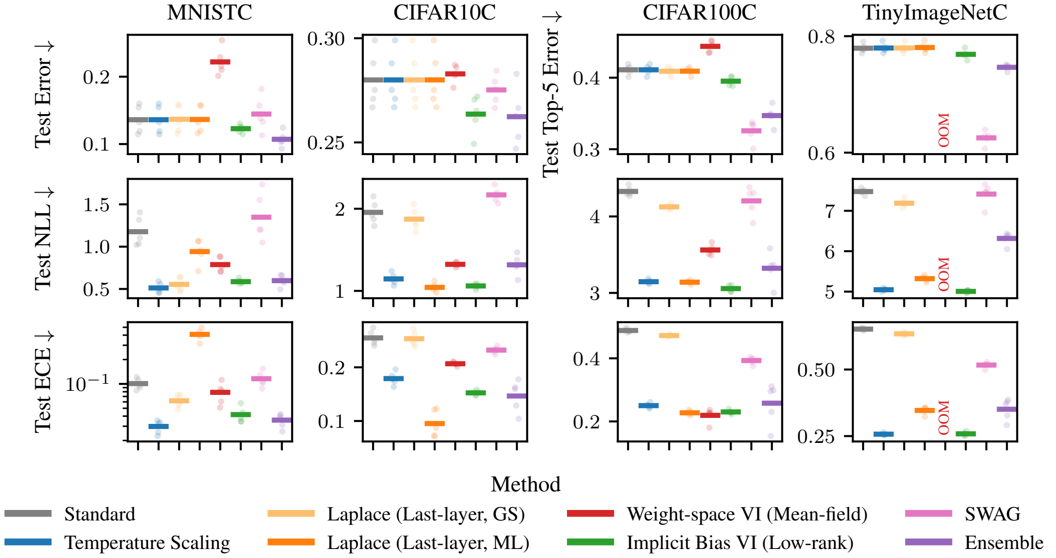


Figure S13: *Generalization on robustness benchmark problems (Maximal Update parametrization).*