A Gaussian Process View on Observation Noise and Initialization in Wide Neural Networks

Sergio Calvo-Ordoñez[*,1,2,3,†] **Jonathan Plenk**[*,1,2] **Richard Bergna**[4] **Álvaro Cartea**[1,2]
Jose Miguel Hernández-Lobato[4] **Konstantina Palla**[5] **Kamil Ciosek**[5]

[1]Mathematical Institute, University of Oxford
[2]Oxford-Man Institute, University of Oxford
[3]OATML, University of Oxford
[4]Department of Engineering, University of Cambridge
[5]Spotify
[*]Equal Contribution, [†]Work partially done during a Spotify internship

Abstract

Performing gradient descent in a wide neural network is equivalent to computing the posterior mean of a Gaussian Process with the Neural Tangent Kernel (NTK-GP), for a specific prior mean and with zero observation noise. However, existing formulations have two limitations: (i) observation noise, since the NTK-GP assumes noiseless targets, leading to misspecification on noisy data; (ii) the equivalence does not extend to arbitrary prior means, which are essential for well-specified models. To address (i), we introduce a regularizer into the training objective, showing its correspondence to incorporating observation noise in the NTK-GP. To address (ii), we propose a *shifted network* that enables arbitrary prior means and allows obtaining the posterior mean with gradient descent on a single network, without ensembling or kernel inversion. We validate our results with experiments across datasets and architectures, showing that this approach removes key obstacles to the practical use of NTK-GP equivalence in applied Gaussian process modeling.

1 INTRODUCTION

The connection between wide neural networks and Gaussian Processes (GPs) via the Neural Tangent Kernel (NTK) (Jacot et al., 2018) provides a powerful framework for understanding the training dynamics of deep learning. The NTK describes how predictions of a neural network evolve under gradient-based optimization in the infinite-width limit and has primarily been used to analyze generalization, convergence, and expressivity of neural networks from a theoretical perspective. While this line of work has yielded insights into training dynamics, its practical utility for GP inference remains limited. In particular, the standard NTK-GP correspondence, as established by Lee et al. (2019), connects the output of a wide neural network trained with gradient flow on mean squared error (MSE) to the posterior mean of a GP with the NTK as its kernel — under restrictive conditions. Specifically, this result assumes (i) zero observation noise, leading to model misspecification when learning from real-world data, and (ii) a fixed prior mean given by a randomly initialized network, which is uninformative and not easily controllable. These constraints make the GP equivalence difficult to leverage in practice, as they limit the model's ability to incorporate uncertainty and prior knowledge, two key concepts of principled Bayesian inference. In this work, we address these limitations and extend the theory in a way that enables practical posterior mean inference with NTK-GPs using a single neural network training run.

Concerning the first limitation, observation — or aleatoric — noise is a fundamental component of Gaussian Processes and other probabilistic models (Williams and Rasmussen, 2006; Sáez de Ocáriz Borde et al., 2024; Ordoñez et al., 2024) that represents observation uncertainty. Measurements in data are subject to precision limits, and labels can carry errors due to annotation noise or ambiguities (Kendall and Gal, 2017). By capturing this variability, aleatoric noise enables models to produce predictions that reflect the randomness of the observed process (Williams and Barber, 1998; Bergna et al., 2024). In GPs, this

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

is achieved through a variance term that accounts for label noise, contributing to robust and well-calibrated predictions.

Hu et al. (2020) tried to address the problem of observation noise by introducing a regularizer penalizing the distance between network parameters at time t, θ_t, and their initialization, θ_0, demonstrating its efficacy in noisy settings. Their analysis, however, relied on the NTK framework and assumed that the network remains in a linear regime throughout training — a critical assumption that was left unproven. This oversight is significant, as the regularizer modifies the training dynamics and invalidates the application of previous results in Lee et al. (2019). Other recent works have used the regularizer proposed by Hu et al. (2020) to improve generalization and training stability. For instance, Nitanda and Suzuki (2020) employed it to constrain networks near their initialization under the NTK regime, while Suh et al. (2021) extended this to deeper networks for recovering ground-truth functions from noisy data. He et al. (2020) explored the regularizer's role in Bayesian deep ensembles.

While these works demonstrate the utility of the regularizer, they all rely on the assumption that wide neural networks remain in a linear regime throughout training. This assumption is critical for leveraging the NTK to characterize the learning dynamics and derive theoretical guarantees, yet the validity of this assumption for the modified gradient flow has not been shown. This gap in the literature raises a key question: how does regularization affect the linearization of wide neural networks, and how does this connect to having a well-specified Bayesian model? In this work, we demonstrate that this regularizer not only preserves the linearity of neural networks but also introduces non-zero aleatoric noise into the NTK-GP mean posterior. We enable the NTK-GP posterior mean to reflect the noise present in real-world data, making it a properly specified Bayesian model. At the same time, this helps justify the generalization properties and benefits of regularization observed in previous literature.

To support inference with arbitrary prior means, we propose the use of a *shifted network* during training. This approach provides a principled strategy to eliminate initialization randomness, ensuring deterministic convergence of a single *shifted network* to the posterior mean of the defined NTK-GP prior.

Our contributions are summarized as follows:

- We demonstrate that regularization in wide neural networks corresponds to aleatoric noise in NTK-GPs, correcting model misspecification.
- We prove that weight-space regularization ensures linear training dynamics under gradient flow and gradient descent.
- We introduce a shifted network framework, enabling arbitrary prior means and deterministic convergence to the NTK-GP posterior mean without ensembling or kernel inversion.
- We validate our theory with experiments on how architecture and regularization affect linearization and convergence.

2 PRELIMINARIES

Parametrizations: Standard versus NTK. The choice of parametrization in neural networks determines how signals propagate and how gradients scale with the network width. Let $\phi : \mathbb{R} \to \mathbb{R}$ denote the activation function[1]. In the standard parametrization, the layer outputs are defined (for $l = 0, \ldots, L$) as:

$$h^{l+1} := W^{l+1}x^l + b^{l+1}, \quad x^{l+1} := \phi(h^{l+1}) \in \mathbb{R}^{n_{l+1}}, \tag{1}$$

where $W^{l+1} \in \mathbb{R}^{n_{l+1} \times n_l}$ and $b^{l+1} \in \mathbb{R}^{n_{l+1}}$ are the weights and biases, respectively. The parameter vector $\theta \in \mathbb{R}^p$ is defined by stacking them. Here, $n_0 := d$ is the dimension of the input $x^0 = x$, and $n_{L+1} := k$ is the dimension of the output $f(x, \theta) := h^{L+1}$. For simplicity we only consider $k = 1$. The initial weights θ_0 are i.i.d. $W^l_{0,ij} \sim \mathcal{N}(0, \frac{\sigma^2_{w,l}}{n_l})$, and the biases as $b^l_{0,i} \sim \mathcal{N}(0, \sigma^2_{b,l})$[2]. Under this parametrization, the norm of the Jacobian diverges for width $n_l \to \infty$. Without loss of generality, we will assume $n_1 = \ldots = n_L =: n$.

In the NTK parametrization, the layer outputs are scaled as:

$$h^{l+1} := \frac{1}{\sqrt{n_l}} W^{l+1}x^l + b^{l+1}, \quad x^{l+1} := \phi(h^{l+1}) \in \mathbb{R}^{n_{l+1}}, \tag{2}$$

where the weights and biases are initialized i.i.d. as $W^l_{0,ij} \sim \mathcal{N}(0, \sigma^2_{w,l})$ and $b^l_{0,i} \sim \mathcal{N}(0, \sigma^2_{b,l})$. This scaling ensures stability in both the forward and backward pass for $n_l \to \infty$.

In Appendix E we will precisely show the equivalence between both parametrizations if one chooses the correct learning rate. Further, we will show that using the same learning rate for each parameter in standard parametrization leads to the first layer and the biases not being trained in the infinite-width limit.

Neural Tangent Kernel. The NTK characterizes the evolution of wide neural network predictions as a

[1] We assume that ϕ and ϕ' are Lipschitz.

[2] Note, that W^1x^0 sums over d terms, and for $l \geq 1$, $W^{l+1}x^l$ sums over n terms.

linear model in function space. Given a neural network $f(x, \theta)$ parameterized by $\theta \in \mathbb{R}^p$, define the parameter-Jacobian in any N points $\mathbf{x}_1, \ldots, \mathbf{x}_N \in \mathbb{R}^d$ as $J(\mathbf{x}, \theta) := \frac{\partial f(\mathbf{x}, \theta)}{\partial \theta} \in \mathbb{R}^{N \times p}$. Under NTK parametrization, the empirical NTK at two sets of inputs $\mathbf{x}'_1, \ldots, \mathbf{x}'_{N'}$ and $\mathbf{x}_1, \ldots, \mathbf{x}_N$ is defined as:

$$\hat{\mathbf{\Theta}}_{\mathbf{x}', \mathbf{x}} := J(\mathbf{x}', \theta_0) J(\mathbf{x}, \theta_0)^\top \in \mathbb{R}^{N' \times N}. \tag{3}$$

The empirical NTK depends on the randomly initialized θ_0. Interestingly, as shown by Jacot et al. (2018), in the infinite width limit, the empirical NTK converges to a deterministic (i.e., independent of θ_0) kernel $\mathbf{\Theta}$ (the analytical NTK) at initialization and remains constant during training (with unregularized gradient flow). A GP defined by the analytical NTK is referred to as the NTK-GP.

3 MOTIVATION

Our work builds on a well-known result (Lee et al., 2019) which shows that, for any sufficiently[3] wide neural network initialized to parameters θ_0, performing gradient descent on the loss

$$\mathcal{L}^0(\theta) := \frac{1}{2} \sum_{i=1}^{N} (f(\mathbf{x}_i, \theta) - \mathbf{y}_i)^2 \tag{4}$$

until convergence gives us a network with predictions arbitrarily close to those given by the closed-form formula

$$f(\mathbf{x}', \theta_0) + \hat{\mathbf{\Theta}}_{\mathbf{x}', \mathbf{x}} \hat{\mathbf{\Theta}}^{-1}_{\mathbf{x}, \mathbf{x}} (\mathbf{y} - f(\mathbf{x}, \theta_0)). \tag{5}$$

This is of special interest because Equation (5) is equivalent to computing the posterior mean of a Gaussian Process with the kernel $\hat{\mathbf{\Theta}}$, zero observation noise, and prior mean $f(\mathbf{x}, \theta_0)$. In other words, there is a clear link between the computationally convenient process of gradient descent and a principled, Bayesian approach to inference.

While the connection between Equations (4) and (5) is interesting, it is by itself not very practical. This is both because we seldom want to do inference for a randomly generated prior mean and because observations in regression tasks are often perturbed by noise. In this paper, we thus ask if it is possible to extend this equivalence by allowing for nonzero observation noise and for prior means other than $f(\mathbf{x}, \theta_0)$. This leads us to the following research question.

[3]In our proofs, we make the notion of 'sufficiently wide' formal.

Main Research Question. Is there a loss function and setting such that performing gradient descent on that loss gives us a network with predictions

$$m(\mathbf{x}') + \hat{\mathbf{\Theta}}_{\mathbf{x}', \mathbf{x}} \left(\hat{\mathbf{\Theta}}_{\mathbf{x}, \mathbf{x}} + \beta I \right)^{-1} (\mathbf{y} - m(\mathbf{x})),$$

for an arbitrary prior mean $m(\mathbf{x})$ and for arbitrary values of observation noise β?

We answer this question in the affirmative. Specifically, Section 4 addresses limitation (i), observation noise, by showing that adding a regularizer to the training objective induces aleatoric noise β in the NTK-GP posterior mean (Theorems 4.1 and 4.3), while Section 5 addresses limitation (ii), arbitrary prior means, via a shifted network construction that enables user-specified priors (Theorem 5.1).

4 OBSERVATION NOISE THROUGH REGULARIZED GRADIENT DESCENT

The introduction of weight-space regularization not only impacts the generalization properties of wide neural networks (Hu et al., 2020), but also fundamentally alters their training dynamics. In the results proposed by Lee et al. (2019) —where the unregularized gradient flow is studied— these dynamics are crucial to understanding how networks converge to solutions that (in the mean posterior sense) align with Bayesian inference. This section addresses limitation (i), observation noise, by analyzing the training dynamics of wide neural networks under regularized gradient flow, with a particular focus on how regularization affects network linearization during training. We demonstrate that the training trajectory stays arbitrarily close to its linearized counterpart, providing both theoretical justification and insights into how regularization modifies the gradient flow.

Define $f(\theta) := f(\mathbf{x}, \theta)$, $g(\theta) := f(\mathbf{x}, \theta) - \mathbf{y} \in \mathbb{R}^N$, $J(\theta) := J(\mathbf{x}, \theta) \in \mathbb{R}^{N \times p}$ in the training points[4]. We will consider (for NTK parametrization[5]) the regularized training loss

$$\mathcal{L}^\beta(\theta) := \frac{1}{2} \sum_{i=1}^{N} (f(\mathbf{x}_i, \theta) - \mathbf{y}_i)^2 + \frac{1}{2} \beta \|\theta - \theta_0\|_2^2 \tag{6}$$

$$= \frac{1}{2} \|g(\theta)\|_2^2 + \frac{1}{2} \beta \|\theta - \theta_0\|_2^2. \tag{7}$$

[4]We assume that $\mathbf{x}_i \neq \mathbf{x}_j$ for $i \neq j$.

[5]See Appendix E.2 for standard parametrization.

The gradient of this loss is given by

$$\nabla_\theta \mathcal{L}^\beta(\theta) = J(\theta)^\top g(\theta) + \beta(\theta - \theta_0). \tag{8}$$

We study the training dynamics under the regularized gradient flow[6][7] $\frac{d\theta_t}{dt} = -\eta \nabla_\theta \mathcal{L}^\beta(\theta_t)$. By the chain rule, $\frac{df(x,\theta_t)}{dt} = J(x,\theta_t)\frac{d\theta_t}{dt}$, and thus the training dynamics of the network are

$$\frac{df(x,\theta_t)}{dt} = -\eta \left(J(x,\theta_t) J(\theta_t)^\top g(\theta_t) + \beta J(x,\theta_t)(\theta_t - \theta_0) \right). \tag{9}$$

For the sake of readability, we omit the dependence of θ_t on θ_0 and β. To gain insights into the role of regularization, we first analyze the regularized gradient flow for the linearized network

$$f^{\text{lin}}_{\theta_0}(x,\theta) := f(x,\theta_0) + J(x,\theta_0)(\theta - \theta_0). \tag{10}$$

This is a linear ODE and hence has a closed-form solution. We formalize this in the theorem below.

Theorem 4.1. *For training time $t \to \infty$, at any point $\mathbf{x}'$,*

$$f^{\text{lin}}_{\theta_0}(\mathbf{x}', \theta_\infty) = f(\mathbf{x}', \theta_0) + \hat{\mathbf{\Theta}}_{\mathbf{x}',\mathbf{x}} \left(\hat{\mathbf{\Theta}}_{\mathbf{x},\mathbf{x}} + \beta I \right)^{-1} (\mathbf{y} - f(\mathbf{x}, \theta_0)). \tag{11}$$

We derive this closed-form solution (for gradient flow and gradient descent) in Appendix D. Note that the statements are high-probability results with respect to the random initialization: for any $\delta_0 > 0$, the probability can be made at least $1 - \delta_0$ by choosing the width n sufficiently large. The required scaling of n with δ_0 is made precise in Appendices A and F.

Moreover, if we consider the network initialization as a random variable, the trained linearized network converges to a normal distribution for layer width $n \to \infty$. Its mean corresponds to the posterior mean of the NTK-GP with prior mean 0. However, similarly to Lee et al. (2019); He et al. (2020), the covariance cannot be interpreted as the posterior covariance of an NTK-GP. See Appendix I for the exact formulas.

In the remainder of this section, we show that, with high probability over initialization, the training dynamics of the network (at any test point $\mathbf{x}'$) closely follow the output of the linearized network for large enough hidden layer width n. First, we leverage the fact that the Jacobian is locally bounded and locally Lipschitz, with a Lipschitz constant of $O\left(\frac{(\log n)^c}{\sqrt{n}}\right)$ (see Appendix A for details). Next, we prove that the distance between the parameters and their initialization $\|\theta_t - \theta_0\|_2$ is $O(1)$[8]. This step is particularly challenging because proofs for $\beta = 0$ (in Eq. (6)) rely on the MSE $\|g(\theta_t)\|_2$ decaying exponentially to 0. As this is not the case for $\beta > 0$, we instead show that $\|\nabla_\theta \mathcal{L}^\beta(\theta_t)\|_2$ decays to 0 exponentially fast.

Even for $\beta = 0$, our analysis provides a generalization of previous proofs in the literature and presents further insights into why the network converges to a global minimizer, despite the Hessian of the loss having negative eigenvalues. Finally, we conclude that the regularized gradient flow remains arbitrarily close to its linearized counterpart, and hence the same holds for the network.

4.1 Exponential Decay of the Regularized Gradient and Parameter Proximity to Initialization

Building upon the established Lipschitz continuity of the Jacobian (in Appendix A), we analyze the effect of regularized gradient flow on the norm of the gradient and the parameters' closeness to their initialization. We begin by demonstrating that the norm of the gradient of the regularized loss decays exponentially over time. This result directly implies that the distance between the parameters and their initialization is bounded by a constant, i.e., $\|\theta_t - \theta_0\|_2 = O(1)$[9]. Furthermore, leveraging the Lipschitzness of the Jacobian established in Lemma A.1, we show that the Jacobian remains close to its initial value, with deviations scaling as $O\left(\frac{(\log n)^c}{\sqrt{n}}\right)$.

Theorem 4.2. *Let $\beta \geq 0$. Let $\delta_0 > 0$ arbitrarily small. There are $K', K, R_0, c_\beta > 0$, such that for n large enough, the following holds with probability of at least $1 - \delta_0$ over random initialization, when applying regularized gradient flow with learning rate $\eta = \eta_0$:*

$$\left\| \frac{d\theta_t}{dt} \right\|_2 = \eta_0 \|\nabla_\theta \mathcal{L}^\beta(\theta_t)\|_2 \leq \eta_0 K R_0 e^{-\eta_0 c_\beta t}, \tag{12}$$

$$\|\theta_t - \theta_0\|_2 \leq \frac{K R_0}{c_\beta} \left(1 - e^{-\eta_0 c_\beta t}\right) < \frac{K R_0}{c_\beta} =: C, \tag{13}$$

$$\forall \|x\|_2 \leq 1 : \|J(x,\theta_t) - J(x,\theta_0)\|_2 \leq \frac{(\log n)^c}{\sqrt{n}} K' C, \tag{14}$$

$$\|J(\theta_t) - J(\theta_0)\|_2 \leq \frac{(\log n)^c}{\sqrt{n}} K C. \tag{15}$$

We provide a proof in Appendix F.1. The main idea is to analyze $\frac{d}{dt}\|\nabla_\theta \mathcal{L}^\beta(\theta_t)\|_2$ and leverage the fact that

[6] In Appendix G, we state equivalent results for gradient descent with small enough learning rates. These will involve geometric sums instead of the exponential.

[7] We will prove that this ODE has a unique solution under our assumptions.

[8] On average, this implies that the distance of individual parameters barely changes.

[9] The bounding constant is independent of the outcome in the high probability event for which this holds.

the eigenvalues of the Hessian of the network scale as $O\left(\frac{(\log n)^c}{\sqrt{n}}\right)$. Consequently, these eigenvalues are dominated by the regularization term $\beta > 0$, leading to the exponential decay. This is a substantial shift from prior proofs for the case $\beta = 0$, where exponential decay is shown for $\|g(\theta_t)\|_2$ (the training error) instead[10].

Our result for $\beta > 0$ does not require the minimum eigenvalue $\lambda_{\min}(\mathbf{\Theta})$ of the analytical NTK to be positive. For the unregularized case $\beta = 0$ (and $\lambda_{\min}(\mathbf{\Theta}) > 0$), we recover the results of Lee et al. (2019) by an alternative proof. This highlights the importance of the gradient being in the row-span of the Jacobian $J(\theta_t)$, ensuring that only the positive eigenvalues of $J(\theta_t)^\top J(\theta_t)$ contribute to the dynamics.

Theorem 4.3. *Let $\beta \geq 0$. Let $\delta_0 > 0$ be arbitrarily small. Then, there are C_1, C_2, such that for n large enough, with probability of at least $1 - \delta_0$ over random initialization,*

$$\sup_{t \geq 0} \|\theta_t - \theta_t^{\text{lin}}\|_2 \leq C_1 \frac{(\log n)^c}{\sqrt{n}}, \tag{16}$$

$$\forall \|x\|_2 \leq 1 : \sup_{t \geq 0} \|f(x, \theta_t) - f_{\theta_0}^{\text{lin}}(x, \theta_t^{\text{lin}})\|_2 \leq C_2 \frac{(\log n)^c}{\sqrt{n}}. \tag{17}$$

θ_t^{lin} are the parameters of the linearized network $f_{\theta_0}^{\text{lin}}$ at time t. See Appendix F.2 for a proof. Again, unlike previous proofs for $\beta = 0$, which rely on the exponential decay of $\|g(\theta_t)\|_2$, we present a more general and simpler approach that applies for $\beta \geq 0$. Specifically, we decompose $\|f(x, \theta_t) - f_{\theta_0}^{\text{lin}}(x, \theta_t^{\text{lin}})\|_2$ into $\|f(x, \theta_t) - f_{\theta_0}^{\text{lin}}(x, \theta_t)\|_2$ and $\|f_{\theta_0}^{\text{lin}}(x, \theta_t) - f_{\theta_0}^{\text{lin}}(x, \theta_t^{\text{lin}})\|_2$, and bound them using Theorem 4.2.

The results in this section establish the key conditions required to conclude that wide neural networks under regularized gradient flow remain in a linear regime throughout training. This enables us to use Theorem 4.1 to analyze network training with the regularized loss.

4.2 NTK-GP Posterior Mean with Aleatoric Noise Interpretation

Given the linearization of our network, we can now look back to Theorem 4.1 and observe that, at convergence under regularized gradient flow and given a random initialization, the output of the linearized network corresponds to the posterior mean of a GP with the analytical NTK as the kernel and non-zero observation noise. This provides a direct Bayesian interpretation of the regularized training process in terms of the NTK-GP framework.

[10]This implies exponential decay of $\|\nabla_\theta \mathcal{L}^0(\theta_t)\|_2 = \|J(\theta_t)^\top g(\theta_t)\|_2$.

It is important to highlight that the trained parameters θ_∞ generally depend on the specific initialization θ_0, and as such, will differ for every random initialization. This variability is relevant to the training process but is not central to our analysis and use case. Unlike deep ensembles (He et al., 2020) (particularly when used in the context of Thompson sampling (Thompson, 1933)), where the distribution of trained networks across initialization is explicitly used, our focus lies on the behaviour of a single trained network under a given initialization. We are able to alleviate the effect of this randomness thanks to our initialization strategy described in the next section.

5 NEURAL NETWORK INITIALIZATION AS NTK-GP WITH ARBITRARY PRIOR MEAN

This section addresses limitation (ii), arbitrary prior means, by introducing a shifted network construction that enables deterministic NTK-GP posterior means with user-specified priors. While standard initialization schemes for neural networks typically involve randomly setting weights and biases to break symmetry, they generally lack a principled way to align the network's prior outputs with specific inductive biases or desired properties. In contrast, Gaussian processes offer a well-defined framework for specifying prior distributions over functions through the prior mean and covariance. Arbitrary initialization is especially valuable in reinforcement learning, where optimistic priors aid exploration (Strehl et al., 2006), and in domains with strong prior knowledge, where encoding a meaningful prior mean can improve performance.

One way to use the results presented in the previous section to obtain the posterior of the NTK-GP with zero prior mean is to train several ensembles and average the outputs using the law of large numbers, akin to the approach by He et al. (2020). However, if we are only interested in computing the posterior mean and not the covariance, there is a better way, which only requires the use of one network. Section 5.1 explores how neural networks, under the NTK-GP framework, can be initialized to reflect arbitrary prior means, enhancing their flexibility to align with specific tasks. Appendix B shows why simply setting the last layer's weights to zero does not yield the desired prior.

5.1 Shifting the labels or predictions

Inspired by standard techniques in GP literature (Williams and Rasmussen, 2006), we provide a formal construction for modifying the network to introduce

arbitrary prior mean. This can be accomplished either by shifting the predictions of the neural network by its predictions at initialization, or equivalently, defining a new shifted network. The intuition behind this is that the shifting does not change the Jacobian of the network with respect to the parameters, but gives a different initialization in Equation (11). The following theorem formalizes this, resolving the main research question we posed in Section 3.

Theorem 5.1. *(Shifted Network.) Consider any function m. Given a random initialization θ_0, define shifted predictions $\tilde{f}_{\theta_0}(\mathbf{x}, \theta)$ as follows:*

$$\tilde{f}_{\theta_0}(\mathbf{x}, \theta) := f(\mathbf{x}, \theta) - f(\mathbf{x}, \theta_0) + m(\mathbf{x}). \tag{18}$$

Training this modified network (starting with θ_0) leads to the following output (in the infinite-width limit)

$$\tilde{f}_{\theta_0}(\mathbf{x}', \theta_\infty) = m(\mathbf{x}') + \mathbf{\Theta}_{\mathbf{x}',\mathbf{x}}(\mathbf{\Theta}_{\mathbf{x},\mathbf{x}} + \beta I)^{-1}(\mathbf{y} - m(\mathbf{x})). \tag{19}$$

This can be interpreted as the posterior mean of an NTK-GP with prior mean function m.

We prove this in Appendix H. Note that $\tilde{f}_{\theta_0}(\mathbf{x}', \theta_\infty)$ is non-random (i.e., independent of the initialization θ_0). This is different from training the non-shifted network, where $f_{\theta_0}(\mathbf{x}', \theta_\infty)$ is random, but has a mean which corresponds to the posterior mean of an NTK-GP with prior mean 0. Note, that the non-randomness of $\tilde{f}_{\theta_0}(\mathbf{x}', \theta_\infty)$ is not immediately clear: while $\tilde{f}_{\theta_0}(x, \theta_0)$ does not depend on θ_0, $\tilde{f}_{\theta_0}(x, \theta)$ differs for different θ_0 when $\theta \neq \theta_0$. The reason for the non-randomness after training is that the NTK doesn't depend on the initialization in the infinite-width limit.

One drawback of shifting predictions is the need to store the initial parameters θ_0, effectively doubling the memory requirements, which can be prohibitive for large networks. Additionally, performing a forward pass of the initial network to compute the shift adds a slight additional computational overhead. However, it is worth noting that no additional back-propagation is required through $f(\mathbf{x}, \theta_0)$ or $m(\mathbf{x})$, keeping the overall cost manageable and lower than even a two-network ensemble.

6 EXPERIMENTS

In this section, we empirically validate our theoretical results by studying the convergence of wide neural networks (and parameters) trained with regularized gradient descent to their linearized counterparts. First, we analyze how the trained parameters deviate from the optimal linearized network parameters (i.e., the converged linear regression weights) and how this difference shrinks as the width increases. We then compare the network's predictions with those of kernel ridge regression on a test set, varying network depth and the regularization strength β. Lastly, we show the effectiveness of using a pre-trained neural network as an informative prior mean in terms of performance across different dataset sizes in a synthetic task.

6.1 Experimental Setup

To ensure alignment with the NTK framework, we train wide fully connected multi-layer perceptrons (MLPs) under the NTK parametrization using full-batch gradient descent (additional results under SGD with different batch sizes are included in Appendix C). The experiments are performed in both synthetic and standard regression datasets. The synthetic problem is a multi-dimensional synthetic regression task where the target function is defined as $\mathbf{y} = \sin(\mathbf{x}) + \cos(2\mathbf{x}) + \boldsymbol{\epsilon}$, with $\boldsymbol{\epsilon} \sim \mathcal{N}(0, I\sigma^2)$ representing the observation noise. Inputs $\mathbf{x}$ are uniformly sampled from the range $[-6, 6]$. We generate a dataset of a few hundred training points and use a two-layer neural network. For the standard regression tasks, we use the datasets *Airline* (Hensman et al., 2013), *Taxi* (Peng et al., 2017; Salimbeni and Deisenroth, 2017), and UCI *Year* (Hernández-Lobato and Adams (2015)), which are complex and high-dimensional. We follow Bergna et al. (2025) for architectural choices and hyperparameter settings. All experiments were performed on a single NVIDIA L40 GPU.

To compare the trained network parameters to their linearized counterparts, we compute the Frobenius norm between the network parameters and those obtained via the kernel ridge regression using the NTK (Section 6.2). Similarly, we evaluate the deviation between the trained neural network and the kernel ridge regression predictions on a test set by computing the squared error (Section 6.3). We repeat these experiments over 5 different seeds and plot the mean and standard error on a logarithmic scale. The plots below in the synthetic experiments are for 3-layer MLPs with $\beta = 0.5$. Results for different depths and β coefficients are shown in linear plots in Appendix C.

Finally, in Section 6.4, we explore how incorporating a learned prior mean can improve data efficiency in transfer settings. We compare training from scratch to initializing with the posterior from a related task, and observe significantly improved performance, especially in low-data regimes.

6.2 Empirical Convergence of the Parameters

We examine how the trained network parameters compare to those obtained via kernel ridge regression with the NTK. Specifically, we compute the ℓ_2 norm dif-

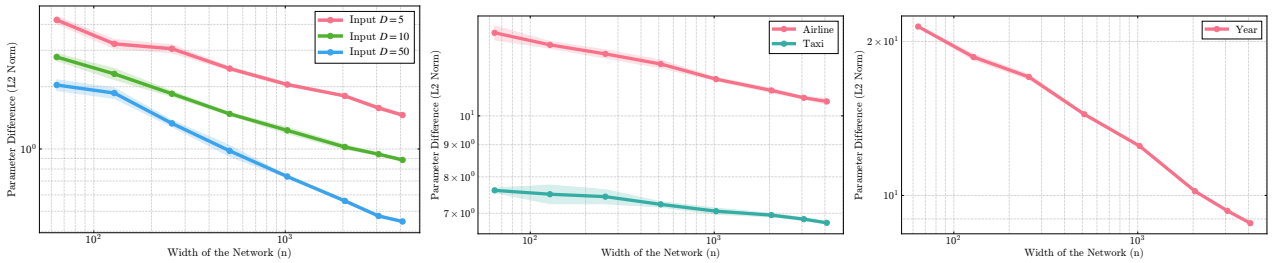

Figure 1: Frobenius norm differences between the trained neural network's parameters and the kernel ridge regression solution plotted against network width for different datasets. **Left:** Different input dimensions of the synthetic dataset. **Middle:** *Airline* and *Taxi* datasets. **Right:** UCI *Year* dataset. Shaded regions represent the standard deviation divided by the square root of the number of seeds.

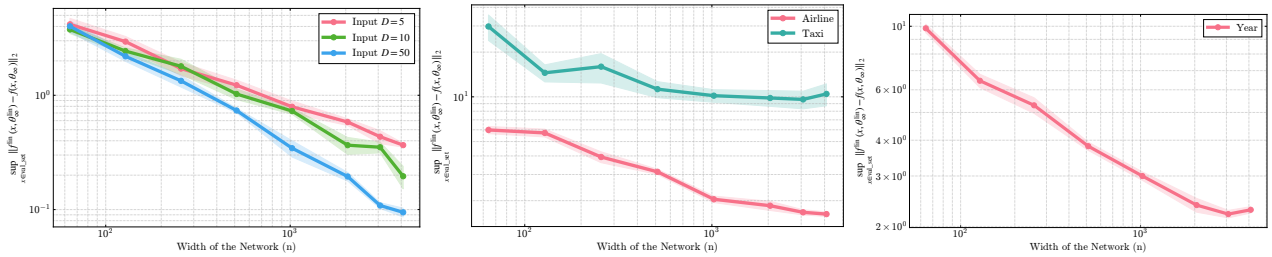

Figure 2: Supremum of the ℓ_2 norm between the trained neural network's outputs $f(x, \theta_\infty)$ and the linearized model's predictions $f^{\text{lin}}(x, \theta_\infty^{\text{lin}})$ across the validation set, plotted against network width. **Left:** Different input dimensions of the synthetic dataset. **Middle:** *Airline* and *Taxi* datasets. **Right:** UCI *Year* dataset.

ference between the final parameters of the trained network and the corresponding optimal linearized solution. Leveraging Theorem 5.1, we set the prior mean to zero for all experiments. Results in Figure 1 confirm that this difference decreases as the network width increases, supporting the theoretical prediction offered in Theorem 4.3.

At smaller widths, we observe a notable deviation between the trained network and the linear regression solution, realizing the effect of finite width. As the width increases, the difference gradually declines, indicating the convergence towards the linearized model post-training in all of the tested datasets. We note that computing and inverting the NTK matrix to compute the reference linear model presented a significant computational bottleneck, making it infeasible to scale to larger widths, depths, or dataset sizes. In contrast, training the neural network was substantially more efficient.

6.3 Empirical Convergence of the Trained Network to a Linear Model

We now shift from parameter convergence to function-space convergence. Here we evaluate how closely the trained neural network approximates the predictions of the corresponding linearized model for unseen data. This validation dataset was sampled from the same data-generating process as the training dataset, comprising 20% of its size. Specifically, we compute $\sup_{x \in \mathcal{V}} \|f(x, \theta_\infty) - f^{\text{lin}}(x, \theta_\infty^{\text{lin}})\|_2$, where $\mathcal{V}$ is the validation set. This represents the distance between the trained network's outputs and the kernel ridge regression solution evaluated in a validation set.

Every plot in Figure 2 shows, under a similar setup as in Section 6.2, that as the width increases, this discrepancy diminishes, providing further empirical confirmation of Theorem 4.3. As in Section 6.2, more plots for different depths can be found in Appendix C.

6.4 Pre-Trained Neural Network as Prior Mean

In this experiment, we construct two N-dimensional regression tasks that share low-frequency structure but differ in their high-frequency components. In both cases, observations follow $\mathbf{y} = f(\mathbf{x}) + \boldsymbol{\epsilon}$, where $\boldsymbol{\epsilon} \sim \mathcal{N}(0, \sigma^2 I)$. Task 1 uses the target function $f_1(\mathbf{x}) = \sin(\mathbf{x}) + 0.5\sin(5\mathbf{x}) + 0.2\sin(20\mathbf{x})$, while Task 2 is defined by $f_2(\mathbf{x}) = \sin(\mathbf{x}) + 0.3\cos(7\mathbf{x}) - 0.2\sin(15\mathbf{x})$. Coefficients were chosen arbitrarily and were not tuned.

We compare two learning strategies: (i) **Vanilla:** initialize a two-layer Softplus network at random and train on Task 2 alone, with zero prior mean. (ii) **Pre-training:** first train the same network on Task 1 to convergence, then use its learned posterior mean as the

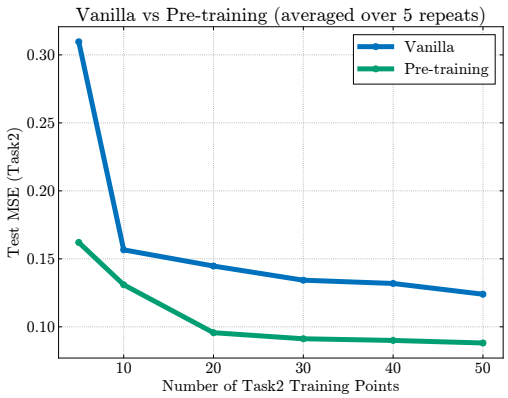

Figure 3: Test MSE on Task 2 as a function of the number of training points, comparing training from scratch (Vanilla) with using the posterior from Task 1 as a prior mean (Pre-training). Pre-training significantly improves performance in low-data regimes.

prior when fitting Task 2. Figure 3 shows the test MSE on Task 2 as a function of n_2. The pre-training curve lies strictly below the vanilla curve—most notably in the low-data regime ($n_2 \leq 20$)—demonstrating that the Task 1 posterior provides a powerful inductive bias that reduces sample complexity on Task 2.

7 RELATED WORK

Linearization Lee et al. (2019) demonstrate that as network width approaches infinity, training dynamics simplify and can be approximated by a linearized model using a first-order Taylor expansion. Lee et al. (2019) also study the links between the output of trained neural networks and GPs. Crucially, we extend this work by proving that this linearization still holds in the presence of observation noise.

Kernel Methods and Neural Networks Jacot et al. (2018) established the equivalence between wide neural networks and kernel methods via kernel ridge regression, and popularized studying neural networks in function space rather than parameter space. We build on both ideas: Theorem 5.1 adopts the function-space view of GPs. Subsequent work has further explored NTK–GP connections in noisy settings. For instance, Rudner et al. (2023) proposed function-space regularization to encode desired properties into predictions, while Chen et al. (2022) related NTK norms to RKHS regularization.

Global Minima and Overparameterization In overparameterized settings, Allen-Zhu et al. (2019) show that SGD can reach global minima in polynomial time, while Zou et al. (2020) prove convergence for ReLU networks by keeping weights close to their initialization. Although we do not directly rely on these results, we also guarantee high-probability convergence to the global optimum.

8 CONCLUSION AND FUTURE WORK

We studied the relationship between regularized training in wide neural networks and Gaussian Processes under the NTK framework. We are the first work formally showing that the proposed weight-space regularization in neural networks is equivalent to adding aleatoric noise to the NTK-GP posterior mean in the infinite-width limit. Additionally, we introduced a shifted network approach that enables arbitrary prior functions and ensures deterministic convergence to the NTK-GP posterior mean without requiring ensembles or kernel inversion. Empirical results validate our theoretical findings, demonstrating convergence to the linearized network. The methods proposed in this paper establish a foundation for practical Gaussian Process inference using the NTK and standard neural network training. Future work could explore the benefits of leveraging the NTK to efficiently train GPs with interpretable and widely used kernels, such as RBF or Matérn. This would require looking into algorithms to compute the posterior covariance (Calvo-Ordoñez et al., 2024; Zanger et al., 2025).

Acknowledgements

SCO's research is supported by the Oxford-Man Institute through the EPSRC Centre for Doctoral Training in Mathematics of Random Systems: Analysis, Modelling and Simulation (EPSRC Grant EP/S023925/1). JP acknowledges financial support from the Oxford-Man Institute. SCO, JP, and AC acknowledge the support of the Oxford-Man Institute for providing computational resources. JMHL acknowledges support from EPSRC funding under grant EP/Y028805/1. JMHL also acknowledges support from a Turing AI Fellowship under grant EP/V023756/1.

References

Arthur Jacot, Franck Gabriel, and Clément Hongler. Neural tangent kernel: Convergence and generalization in neural networks. *Advances in neural information processing systems*, 31, 2018.

Jaehoon Lee, Lechao Xiao, Samuel Schoenholz, Yasaman Bahri, Roman Novak, Jascha Sohl-Dickstein, and Jeffrey Pennington. Wide neural networks of any depth evolve as linear models under gradient descent. *Advances in neural information processing systems*, 32, 2019.

Christopher KI Williams and Carl Edward Rasmussen.

Gaussian processes for machine learning, volume 2. MIT press Cambridge, MA, 2006.

Haitz Sáez de Ocáriz Borde, Alvaro Arroyo, Ismael Morales, Ingmar Posner, and Xiaowen Dong. Neural latent geometry search: product manifold inference via gromov-hausdorff-informed bayesian optimization. *Advances in Neural Information Processing Systems*, 36, 2024.

Sergio Calvo Ordoñez, Matthieu Meunier, Francesco Piatti, and Yuantao Shi. Partially stochastic infinitely deep bayesian neural networks. In *Forty-first International Conference on Machine Learning*, 2024.

Alex Kendall and Yarin Gal. What uncertainties do we need in bayesian deep learning for computer vision? *Advances in neural information processing systems*, 30, 2017.

Christopher KI Williams and David Barber. Bayesian classification with gaussian processes. *IEEE Transactions on pattern analysis and machine intelligence*, 20(12):1342–1351, 1998.

Richard Bergna, Sergio Calvo-Ordonez, Felix L Opolka, Pietro Liò, and Jose Miguel Hernandez-Lobato. Uncertainty modeling in graph neural networks via stochastic differential equations. *arXiv preprint arXiv:2408.16115*, 2024.

Wei Hu, Zhiyuan Li, and Dingli Yu. Simple and effective regularization methods for training on noisily labeled data with generalization guarantee, 2020. URL `https://arxiv.org/abs/1905.11368`.

Atsushi Nitanda and Taiji Suzuki. Optimal rates for averaged stochastic gradient descent under neural tangent kernel regime. *arXiv preprint arXiv:2006.12297*, 2020.

Namjoon Suh, Hyunouk Ko, and Xiaoming Huo. A nonparametric regression viewpoint: Generalization of overparametrized deep relu network under noisy observations. In *International Conference on Learning Representations*, 2021.

Bobby He, Balaji Lakshminarayanan, and Yee Whye Teh. Bayesian deep ensembles via the neural tangent kernel. *Advances in neural information processing systems*, 33: 1010–1022, 2020.

William R. Thompson. On the likelihood that one unknown probability exceeds another in view of the evidence of two samples. *Biometrika*, 25 (3/4):285–294, 1933. ISSN 00063444. URL `http://www.jstor.org/stable/2332286`.

Alexander L Strehl, Lihong Li, Eric Wiewiora, John Langford, and Michael L Littman. Pac model-free reinforcement learning. In *Proceedings of the 23rd international conference on Machine learning*, pages 881–888, 2006.

James Hensman, Nicolo Fusi, and Neil D Lawrence. Gaussian processes for big data. *arXiv preprint arXiv:1309.6835*, 2013.

Hao Peng, Shandian Zhe, Xiao Zhang, and Yuan Qi. Asynchronous distributed variational gaussian process for regression. In *International Conference on Machine Learning*, pages 2788–2797. PMLR, 2017.

Hugh Salimbeni and Marc Deisenroth. Doubly stochastic variational inference for deep gaussian processes. *Advances in neural information processing systems*, 30, 2017.

José Miguel Hernández-Lobato and Ryan Adams. Probabilistic backpropagation for scalable learning of bayesian neural networks. In *International conference on machine learning*, pages 1861–1869. PMLR, 2015.

Richard Bergna, Stefan Depeweg, Sergio Calvo Ordonez, Jonathan Plenk, Alvaro Cartea, and Jose Miguel Hernandez-Lobato. Post-hoc uncertainty quantification in pre-trained neural networks via activation-level gaussian processes. *arXiv preprint arXiv:2502.20966*, 2025.

Tim G. J. Rudner, Sanyam Kapoor, Shikai Qiu, and Andrew Gordon Wilson. Function-space regularization in neural networks: A probabilistic perspective, 2023. URL `https://arxiv.org/abs/2312.17162`.

Zonghao Chen, Xupeng Shi, Tim GJ Rudner, Qixuan Feng, Weizhong Zhang, and Tong Zhang. A neural tangent kernel perspective on function-space regularization in neural networks. In *OPT 2022: Optimization for Machine Learning (NeurIPS 2022 Workshop)*, 2022.

Zeyuan Allen-Zhu, Yuanzhi Li, and Zhao Song. A convergence theory for deep learning via over-parameterization. In *International conference on machine learning*, pages 242–252. PMLR, 2019.

Difan Zou, Yuan Cao, Dongruo Zhou, and Quanquan Gu. Gradient descent optimizes over-parameterized deep relu networks. *Machine learning*, 109:467–492, 2020.

Sergio Calvo-Ordoñez, Konstantina Palla, and Kamil Ciosek. Epistemic uncertainty and observation noise with the neural tangent kernel. *arXiv preprint arXiv:2409.03953*, 2024.

Moritz A. Zanger, Pascal R. Van der Vaart, Wendelin Böhmer, and Matthijs T. J. Spaan. Contextual similarity distillation: Ensemble uncertainties with a single model, 2025. URL `https://arxiv.org/abs/2503.11339`.

Chaoyue Liu, Libin Zhu, and Misha Belkin. On the linearity of large non-linear models: when and why the tangent kernel is constant. *Advances in Neural Information Processing Systems*, 33:15954–15964, 2020.

Jaehoon Lee, Yasaman Bahri, Roman Novak, Samuel S Schoenholz, Jeffrey Pennington, and Jascha Sohl-Dickstein. Deep neural networks as gaussian processes. *International Conference on Learning Representations*, 37, 2018.

Greg Yang. Tensor programs ii: Neural tangent kernel for any architecture. *arXiv preprint arXiv:2006.14548*, 2020.

Aitor Lewkowycz and Guy Gur-Ari. On the training dynamics of deep networks with l_2 regularization. *Advances in Neural Information Processing Systems*, 33: 4790–4799, 2020.

Checklist

1. For all models and algorithms presented, check if you include:
   (a) A clear description of the mathematical setting, assumptions, algorithm, and/or model. Yes.

(b) An analysis of the properties and complexity (time, space, sample size) of any algorithm. Yes.

(c) (Optional) Anonymized source code, with specification of all dependencies, including external libraries. We will provide this upon acceptance.

2. For any theoretical claim, check if you include:

(a) Statements of the full set of assumptions of all theoretical results. Yes.

(b) Complete proofs of all theoretical results. Yes.

(c) Clear explanations of any assumptions. Yes.

3. For all figures and tables that present empirical results, check if you include:

(a) The code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL). We will provide the code upon acceptance. The data and instructions are provided.

(b) All the training details (e.g., data splits, hyperparameters, how they were chosen). Yes.

(c) A clear definition of the specific measure or statistics and error bars (e.g., with respect to the random seed after running experiments multiple times). Yes.

(d) A description of the computing infrastructure used. (e.g., type of GPUs, internal cluster, or cloud provider). Yes.

4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets, check if you include:

(a) Citations of the creator If your work uses existing assets. Yes.

(b) The license information of the assets, if applicable. Not Applicable.

(c) New assets either in the supplemental material or as a URL, if applicable. Not Applicable.

(d) Information about consent from data providers/curators. Not Applicable.

(e) Discussion of sensible content if applicable, e.g., personally identifiable information or offensive content. Not Applicable.

5. If you used crowdsourcing or conducted research with human subjects, check if you include:

(a) The full text of instructions given to participants and screenshots. Not Applicable.

(b) Descriptions of potential participant risks, with links to Institutional Review Board (IRB) approvals if applicable. Not Applicable.

(c) The estimated hourly wage paid to participants and the total amount spent on participant compensation. Not Applicable.

A LOCAL BOUNDEDNESS AND LIPSCHITZNESS OF THE JACOBIAN

To analyze the training dynamics of wide neural networks, it is essential to ensure that the Jacobian of the network remains (locally) bounded and Lipschitz-continuous during training, with the constants scaling "well" with layer width n. This is shown by the following Lemma, which was proved in Lee et al. (2019) for standard parametrization, and more formally in Liu et al. (2020) for NTK parameterization.

Lemma A.1. *For any $\delta_0 > 0$, there is $K' > 0$ (independent of C), such that: For every $C > 0$, there is n large enough, such that with probability of at least $1 - \delta_0$ (over random initialization): For any point x with $\|x\|_2 \leq 1$:*

$$\forall \theta \in B(\theta_0, C) : \|J(x, \theta)\|_2 \leq K', \tag{20}$$

$$\forall \theta, \tilde{\theta} \in B(\theta_0, C) : \|J(x, \theta) - J(x, \tilde{\theta})\|_2 \leq \frac{(\log n)^c}{\sqrt{n}} K' \|\theta - \tilde{\theta}\|_2. \tag{21}$$

In particular, with $K = \sqrt{N}K'$, for the Jacobian over the training points:

$$\forall \theta \in B(\theta_0, C) : \|J(\theta)\|_F \leq K, \tag{22}$$

$$\forall \theta, \tilde{\theta} \in B(\theta_0, C) : \|J(\theta) - J(\tilde{\theta})\|_F \leq \frac{(\log n)^c}{\sqrt{n}} K \|\theta - \tilde{\theta}\|_2. \tag{23}$$

As a direct consequence, for the Hessian $\nabla^2_\theta f(x, \theta) \in \mathbb{R}^{p \times p}$ of the network,

$$\forall \theta \in B(\theta_0, C) : \left\|\nabla^2_\theta f(x, \theta)\right\|_2 \leq \frac{(\log n)^c}{\sqrt{n}} K'. \tag{24}$$

The Frobenius norm is used to aggregate over different training points, i.e. $\|J(\theta)\|_F^2 = \sum_{i=1}^N \|J(\mathbf{x}_i, \theta)\|_2^2$ for $J(\theta) = J(\mathbf{x}, \theta) \in \mathbb{R}^{N \times p}$. The logarithmic term $(\log n)^c$ is sometimes omitted in the literature. Its power $c > 0$ is a constant that only depends on the network architecture.

B INITIALIZING THE LAST LAYER WEIGHTS AS 0 DOES NOT LEAD TO A ZERO PRIOR MEAN NTK-GP

In this section, we explore a tempting but ultimately futile way of constructing a network with zero prior mean. One might be tempted to just initialize the last layer of the network to 0, i.e., $\sigma_{w,L+1} = 0$, $\sigma_{b,L+1} = 0$. Then, the gradient with respect to any weights/biases not in the last layer will be 0. In this case, the empirical NTK is given by

$$\hat{\mathbf{\Theta}}_{\mathbf{x}',\mathbf{x}} = \sum_{l=1}^{L+1} J(\mathbf{x}', \theta^l) J(\mathbf{x}, \theta^l)^\top = J(\mathbf{x}', \theta^{L+1}) J(\mathbf{x}, \theta^{L+1})^\top, \tag{25}$$

where θ^l indicates the weights at layer l. We have,

$$J(\mathbf{x}, W^{L+1}) = \frac{1}{\sqrt{n_L}} x^L(\mathbf{x}, \theta), \quad J(\mathbf{x}, b^{L+1}) = 1. \tag{26}$$

Thus,

$$\hat{\mathbf{\Theta}}_{\mathbf{x}',\mathbf{x}} = \frac{1}{n} x^L(\mathbf{x}', \theta) x^L(\mathbf{x}, \theta)^\top + 1. \tag{27}$$

This actually converges to the Neural Network Gaussian Process (NNGP) kernel (Lee et al., 2018), which does not align with the desired result.

However, our observation highlights that training the network in this configuration corresponds to performing inference in a Gaussian Process (GP) with a zero prior mean and the NNGP kernel as the covariance function. Specifically, in the case of the linearized network (or a network with a sufficiently large layer width), the partial derivatives with respect to the hidden layer parameters are zero. Thus, effectively *only the last layer is trained*. As shown in Lee et al. (2019), this scenario is equivalent to performing inference with the NNGP, where the covariance of the trained network matches the posterior covariance of the NNGP.

C ADDITIONAL EXPERIMENTS

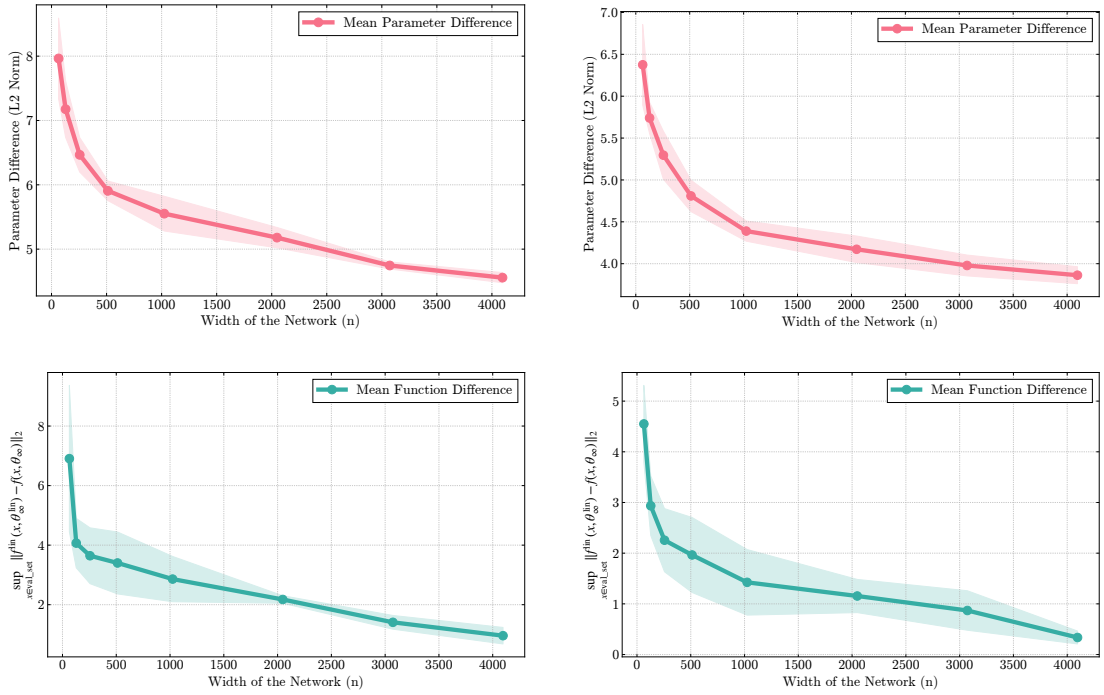

Figure 4: Parameter and function differences for additional network depths. (Top row) Parameter difference plots from Section 6.2. (Bottom row) corresponds to the function difference plots from Section 6.3. (Left) Results for one fully connected hidden layer. (Right) Results for an MLP with three fully connected hidden layers. In all cases, increasing the network width reduces both parameter and function differences, confirming the theoretical predictions. $\beta = 0.1$ was used.

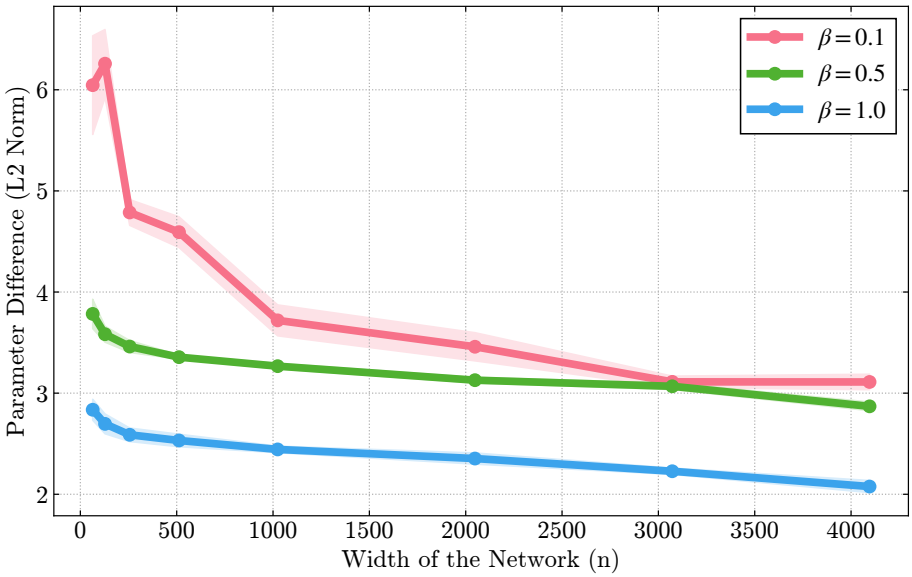

Figure 5: ℓ_2 norm differences between the trained neural network's parameters and the kernel ridge regression solution plotted against network width for different β. Shaded regions represent the standard deviation divided by the square root of the number of seeds.

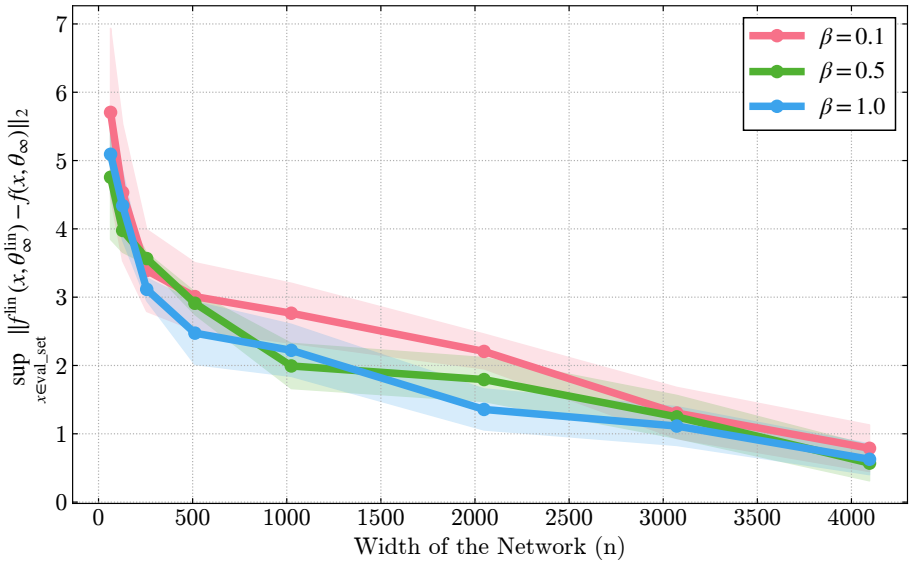

Figure 6: Supremum of the ℓ_2 norm differences between the trained neural network's outputs $f(x, \theta_\infty)$ and the linearized model's predictions $f^{\text{lin}}(x, \theta_\infty^{\text{lin}})$ across the validation set, plotted against network width.

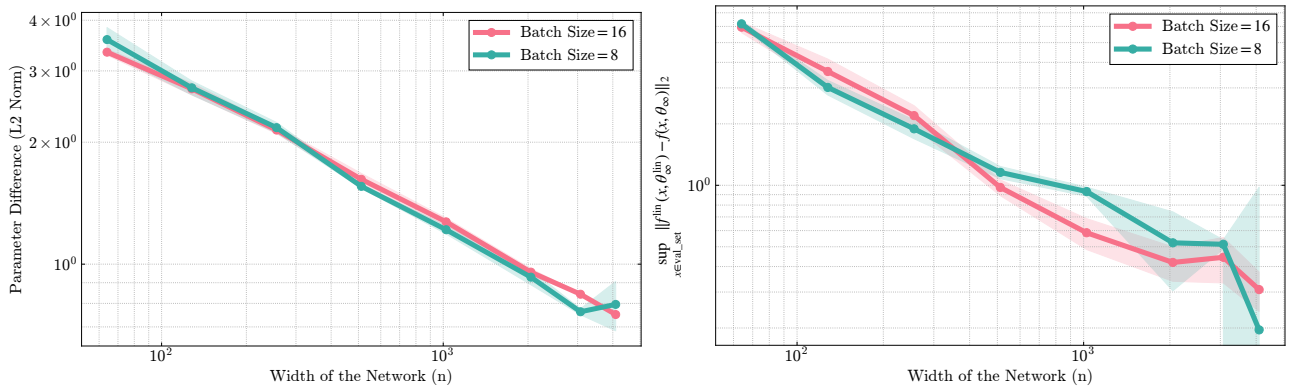

Figure 7: Parameter differences (left) and function differences (right) on the synthetic dataset using mini-batch stochastic gradient descent. Each curve corresponds to a different batch size.

Method	**Test RMSE ↓**
Exact Matern GP	OOM
Inducing Points Matern GP (200)	0.075
NTK-GP via SGD (ours)	0.099
Ensemble of 5 Wide MLPs	0.104
Wide MLP (no regularization)	0.788
Shallow MLP	0.201

Table 1: Comparison on the Kin8nm regression dataset. We compare our method (NTK-GP posterior mean via SGD with zero prior mean) against exact and approximate Matern GPs, ensembles of wide MLPs (He et al., 2020), an unregularized wide MLP (Lee et al., 2019), and a shallow MLP. Exact GP inference resulted in out-of-memory (OOM) issues. Our method outperforms the ensemble while requiring only a single training run and provides a scalable approximation to the GP posterior mean.

D REGULARIZED GRADIENT FLOW AND GRADIENT DESCENT FOR THE LINEARIZED NETWORK

Consider the linearized network $f^{\text{lin}}_{\theta_0}(x,\theta) = f(x,\theta_0) + J(x,\theta_0)(\theta-\theta_0)$. For notational convenience, we drop the dependence on θ_0 throughout the Appendix. In the following, we will consider training the parameters using the regularized training loss

$$\mathcal{L}^{\beta,\text{lin}}(\theta) := \frac{1}{2}\sum_{i=1}^{N}(f^{\text{lin}}(\mathbf{x}_i,\theta)-\mathbf{y}_i)^2 + \frac{1}{2}\beta\|\theta-\theta_0\|_2^2. \tag{28}$$

D.1 Regularized gradient flow for the linearized network

Then, the evolution of the parameters through gradient flow with learning rate η_0 is given by

$$\begin{aligned}
\frac{d\theta_t^{\text{lin}}}{dt} &= -\eta_0\left(J(\theta_0)^\top g^{\text{lin}}(\theta_t^{\text{lin}}) + \beta(\theta_t^{\text{lin}}-\theta_0)\right) && (29)\\
&= -\eta_0\left(J(\theta_0)^\top\left(f(\theta_0)+J(\theta_0)(\theta_t^{\text{lin}}-\theta_0)-\mathbf{y}\right)+\beta(\theta_t^{\text{lin}}-\theta_0)\right) && (30)\\
&= -\eta_0\left(J(\theta_0)^\top J(\theta_0)+\beta I_p\right)(\theta_t^{\text{lin}}-\theta_0) - \eta_0 J(\theta_0)^\top(f(\theta_0)-\mathbf{y}). && (31)
\end{aligned}$$

This is a multidimensional linear ODE in $\theta_t^{\text{lin}}-\theta_0$. Its unique solution is given by

$$\begin{aligned}
\theta_t^{\text{lin}} &= \theta_0 + \left(e^{-\eta_0\left(J(\theta_0)^\top J(\theta_0)+\beta I_p\right)t} - I_p\right)\left(-\eta_0\left(J(\theta_0)^\top J(\theta_0)+\beta I_p\right)\right)^{-1}\left(-\eta_0 J(\theta_0)^\top(f(\theta_0)-\mathbf{y})\right) && (32)\\
&= \theta_0 + \left(I_p - e^{-\eta_0\left(J(\theta_0)^\top J(\theta_0)+\beta I_p\right)t}\right)\left(J(\theta_0)^\top J(\theta_0)+\beta I_p\right)^{-1} J(\theta_0)^\top(\mathbf{y}-f(\theta_0)) && (33)\\
&= \theta_0 + \left(I_p - e^{-\eta_0\left(J(\theta_0)^\top J(\theta_0)+\beta I_p\right)t}\right) J(\theta_0)^\top\left(J(\theta_0)J(\theta_0)^\top+\beta I_N\right)^{-1}(\mathbf{y}-f(\theta_0)) && (34)\\
&= \theta_0 + \left(J(\theta_0)^\top - e^{-\eta_0\left(J(\theta_0)^\top J(\theta_0)+\beta I_p\right)t} J(\theta_0)^\top\right)\left(J(\theta_0)J(\theta_0)^\top+\beta I_N\right)^{-1}(\mathbf{y}-f(\theta_0)) && (35)\\
&= \theta_0 + J(\theta_0)^\top\left(I_N - e^{-\eta_0\left(J(\theta_0)J(\theta_0)^\top+\beta I_N\right)t}\right)\left(J(\theta_0)J(\theta_0)^\top+\beta I_N\right)^{-1}(\mathbf{y}-f(\theta_0)). && (36)
\end{aligned}$$

In the third and fourth equality, we used that for $k\in\mathbb{Z}$,

$$\left(J(\theta_0)^\top J(\theta_0)+\beta I_p\right)^k J(\theta_0)^\top = J(\theta_0)^\top\left(J(\theta_0)J(\theta_0)^\top+\beta I_N\right)^k. \tag{37}$$

Plugging θ_t^{lin} into the formula for the linearized network, we get for any point $\mathbf{x}'$,

$$\begin{aligned}
& f^{\text{lin}}(\mathbf{x}',\theta_t^{\text{lin}}) && (38)\\
=& f(\mathbf{x}',\theta_0) + J(\mathbf{x}',\theta_0)J(\theta_0)^\top\left(I_N - e^{-\eta_0\left(J(\theta_0)J(\theta_0)^\top+\beta I_N\right)t}\right)\left(J(\theta_0)J(\theta_0)^\top+\beta I_N\right)^{-1}(\mathbf{y}-f(\theta_0)) && (39)\\
=& f(\mathbf{x}',\theta_0) + \hat{\mathbf{\Theta}}_{\mathbf{x}',\mathbf{x}}\left(I_N - e^{-\eta_0(\hat{\mathbf{\Theta}}_{\mathbf{x},\mathbf{x}}+\beta I_N)t}\right)\left(\hat{\mathbf{\Theta}}_{\mathbf{x},\mathbf{x}}+\beta I_N\right)^{-1}(\mathbf{y}-f(\theta_0)). && (40)
\end{aligned}$$

For training time $t\to\infty$, this gives

$$f^{\text{lin}}(\mathbf{x}',\theta_\infty^{\text{lin}}) = f(\mathbf{x}',\theta_0) + \hat{\mathbf{\Theta}}_{\mathbf{x}',\mathbf{x}}\left(\hat{\mathbf{\Theta}}_{\mathbf{x},\mathbf{x}}+\beta I_N\right)^{-1}(\mathbf{y}-f(\theta_0)). \tag{41}$$

D.2 Regularized gradient descent for the linearized network

Similarly to gradient flow, the evolution of the parameters through gradient descent (when training the regularized loss given by the linearized network) with learning rate η_0 is given by

$$\theta_t^{\text{lin}} = \theta_{t-1}^{\text{lin}} - \eta_0\left(J(\theta_0)^\top J(\theta_0)+\beta I_p\right)(\theta_{t-1}^{\text{lin}}-\theta_0) - \eta_0 J(\theta_0)^\top(f(\theta_0)-\mathbf{y}). \tag{42}$$

One may write this as

$$\theta_t^{\text{lin}} - \theta_0 = \left(I_p - \eta_0\left(J(\theta_0)^\top J(\theta_0)+\beta I_p\right)\right)(\theta_{t-1}^{\text{lin}}-\theta_0) - \eta_0 J(\theta_0)^\top(f(\theta_0)-\mathbf{y}). \tag{43}$$

Applying the formula for $\theta_t^{\text{lin}} - \theta_0$ iteratively, leads to the following geometric sum:

$$\theta_t^{\text{lin}} - \theta_0 = -\eta_0 \sum_{u=0}^{t-1} \left(I_p - \eta_0 \left(J(\theta_0)^\top J(\theta_0) + \beta I_p\right)\right)^u J(\theta_0)^\top (f(\theta_0) - \mathbf{y}) \tag{44}$$

$$= \eta_0 J(\theta_0)^\top \sum_{u=0}^{t-1} \left(I_N - \eta_0 \left(J(\theta_0) J(\theta_0)^\top + \beta I_N\right)\right)^u (\mathbf{y} - f(\theta_0)) \tag{45}$$

$$= J(\theta_0)^\top \left(I_N - \left(I_N - \eta_0 \left(J(\theta_0) J(\theta_0)^\top + \beta I_N\right)\right)^t\right) \left(J(\theta_0) J(\theta_0)^\top + \beta I_N\right)^{-1} (\mathbf{y} - f(\theta_0)). \tag{46}$$

This converges for $t \to \infty$ if and only if $0 < \eta_0 < \frac{2}{\lambda_{\max}(J(\theta_0)J(\theta_0)^\top) + \beta}$. In that case, it converges (as expected) to the same $\theta_\infty^{\text{lin}}$ as the regularized gradient flow. Plugging θ_t^{lin} into the formula for the linearized network, we get for any point $\mathbf{x}'$,

$$f^{\text{lin}}(\mathbf{x}', \theta_t^{\text{lin}}) \tag{47}$$

$$= f(\mathbf{x}', \theta_0) + J(\mathbf{x}', \theta_0) J(\theta_0)^\top \left(I_N - \left(I_N - \eta_0 \left(J(\theta_0) J(\theta_0)^\top + \beta I_N\right)\right)^t\right) \left(J(\theta_0) J(\theta_0)^\top + \beta I_N\right)^{-1} (\mathbf{y} - f(\theta_0)) \tag{48}$$

$$= f(\mathbf{x}', \theta_0) + \hat{\mathbf{\Theta}}_{\mathbf{x}', \mathbf{x}} \left(I_N - \left(I_N - \eta_0 \left(\hat{\mathbf{\Theta}}_{\mathbf{x}, \mathbf{x}} + \beta I_N\right)\right)^t\right) \left(\hat{\mathbf{\Theta}}_{\mathbf{x}, \mathbf{x}} + \beta I_N\right)^{-1} (\mathbf{y} - f(\theta_0)). \tag{49}$$

E REVISITING STANDARD AND NTK PARAMETRIZATIONS, AND CONVERGENCE AT INITIALIZATION

In the following, we revisit the standard and the NTK parametrization. First, we repeat the result about the convergence of the NTK for the NTK parametrization at initialization. Then, we formally state how the NTK and standard parametrization are related, which makes it possible to proof the results for standard parametrization by using the results for NTK parametrization. Finally, we argue that using the same learning rate for every parameter under standard parametrization leads to redundancies in the NTK for the first layer and the biases.

E.1 Convergence of NTK under NTK parametrization at initialization

Here, we restate the following Theorem from Yang (2020) about the convergence of the NTK at initialization. This was first shown in Jacot et al. (2018) when taking the limit of layer widths sequentially.

Theorem E.1. *Consider a standard feedforward neural network in NTK parametrization. Then, the empirical NTK $\hat{\mathbf{\Theta}}_{\mathbf{x}', \mathbf{x}}$ converges to a deterministic matrix $\mathbf{\Theta}_{\mathbf{x}', \mathbf{x}}$, which we call the analytical NTK:*

$$\hat{\mathbf{\Theta}}_{x', x} = J(x', \theta_0) J(x, \theta_0)^\top = \sum_{l=1}^{L+1} \left(J(x', W^l) J(x, W^l)^\top + J(x', b^l) J(x, b^l)^\top\right) \xrightarrow{p} \mathbf{\Theta}_{x', x}, \tag{50}$$

for layer width $n \to \infty$.

Define $\mathbf{\Theta} := \mathbf{\Theta}_{\mathbf{x}, \mathbf{x}} \in \mathbb{R}^{N \times N}$ as the analytical NTK on the training points. We will assume $\lambda_{\min}(\mathbf{\Theta}) > 0$. A sufficient condition for this is that $\|\mathbf{x}_i\|_2 = 1 \; \forall i$, and that ϕ grows non-polynomially for large x, see Jacot et al. (2018). This directly implies that for n large enough, the minimum eigenvalue of the analytical NTK is lower bounded by a positive number with high probability:

Lemma E.2. *For any $\delta_0 > 0$, there is n large enough, such that with probability of at least $1 - \delta_0$, for the minimum eigenvalue of the empirical NTK,*

$$\lambda_{\min}\left(J(\theta_0) J(\theta_0)^\top\right) \geq \frac{1}{2} \lambda_{\min}(\mathbf{\Theta}), \text{ and } \lambda_{\max}\left(J(\theta_0) J(\theta_0)^\top\right) \leq 2 \lambda_{\max}(\mathbf{\Theta}). \tag{51}$$

E.2 Equivalence of NTK parametrization to standard parametrization with layer-dependent learning rates

In this section, we will formally show how the NTK parametrization relates to the standard parametrization of neural networks. This makes it possible to prove results for standard parametrization by using the results for NTK parametrization, instead of having to prove them again.

Remember that the number of parameters is $p = \sum_{l=1}^{L+1}(n_{l-1}+1)n_l$. Define the diagonal matrix $H \in \mathbb{R}^{p\times p}$ through

$$H := \operatorname{diag}(H_{w,1}, H_{b,1}, \ldots, H_{w,L+1}, H_{b,L+1}), \tag{52}$$

where $H_{w,l} := \frac{1}{n_{l-1}} I_{n_{l-1}n_l}$, and $H_{b,l} := I_{n_l}$. The diagonal of $H^{\frac{1}{2}}$ contains the scalars by which each parameter is multiplied when going from NTK parametrization to standard parametrization. For θ_0^{std} initialized in standard parametrization, define

$$\theta_0^{\text{ntk}} := H^{-\frac{1}{2}}\theta_0^{\text{std}}. \tag{53}$$

Then, θ_0^{ntk} is initialized as in NTK parametrization. Further, let f^{std} denote a neural network in standard parametrization. Then,

$$f^{\text{ntk}}(x, \theta^{\text{ntk}}) := f^{\text{std}}(x, H^{\frac{1}{2}}\theta^{\text{ntk}}), \tag{54}$$

defines a neural network in NTK parametrization. Differentiating gives

$$J^{\text{ntk}}(x, \theta^{\text{ntk}}) = J^{\text{std}}(x, H^{\frac{1}{2}}\theta^{\text{ntk}})H^{\frac{1}{2}}, \quad \hat{\mathbf{\Theta}}^{\text{ntk}}_{x',x} = J^{\text{std}}(x', H^{\frac{1}{2}}\theta^{\text{ntk}})HJ^{\text{std}}(x, H^{\frac{1}{2}}\theta^{\text{ntk}}). \tag{55}$$

Motivated by this, define the following regularized loss:

$$\mathcal{L}^{\beta,\text{std}}(\theta) := \frac{1}{2}\sum_{i=1}^{N}(f^{\text{std}}(\mathbf{x}_i, \theta) - \mathbf{y}_i)^2 + \frac{1}{2}\beta(\theta-\theta_0)^\top H^{-1}(\theta-\theta_0). \tag{56}$$

The resulting gradient is

$$\nabla_\theta \mathcal{L}^{\beta,\text{std}}(\theta) = J^{\text{std}}(\theta)^\top g^{\text{std}}(\theta) + \beta H^{-1}(\theta-\theta_0). \tag{57}$$

Define θ_t^{std} as parameters evolving by gradient flow of the regularized objective in standard parametrization[11], with layer-dependent learning rate $\eta_0 H$:

$$\frac{d\theta_t^{\text{std}}}{dt} = -\eta_0 H \nabla_\theta \mathcal{L}^{\beta,\text{std}}(\theta_t^{\text{std}}) = -\eta_0 H J^{\text{std}}(\theta_t^{\text{std}})^\top g^{\text{std}}(\theta_t^{\text{std}}) - \eta_0\beta(\theta_t^{\text{std}} - \theta_0^{\text{std}}). \tag{58}$$

We define $\theta_t^{\text{ntk}} := H^{-\frac{1}{2}}\theta_t^{\text{std}}$. Then, as $\frac{d\theta_t^{\text{ntk}}}{dt} = H^{-\frac{1}{2}}\frac{d\theta_t^{\text{std}}}{dt}$,

$$\begin{aligned}
\frac{d\theta_t^{\text{ntk}}}{dt} &= -\eta_0 H^{\frac{1}{2}} J^{\text{std}}(\theta_t^{\text{std}})^\top g^{\text{std}}(\theta_t^{\text{std}}) - \eta_0 H^{-\frac{1}{2}}\beta(\theta_t^{\text{std}} - \theta_0^{\text{std}}) && (59)\\
&= -\eta_0 \left(J^{\text{std}}(H^{\frac{1}{2}}\theta_t^{\text{ntk}})H^{\frac{1}{2}}\right)^\top g^{\text{std}}(H^{\frac{1}{2}}\theta_t^{\text{ntk}}) - \eta_0\beta(\theta_t^{\text{ntk}} - \theta_0^{\text{ntk}}) && (60)\\
&= -\eta_0 J^{\text{ntk}}(\theta_t^{\text{ntk}})^\top g^{\text{ntk}}(\theta_t^{\text{ntk}}) - \eta_0\beta(\theta_t^{\text{ntk}} - \theta_0^{\text{ntk}}). && (61)
\end{aligned}$$

Thus, θ_t^{ntk} follows the regularized gradient flow of the objective under NTK parametrization with learning rate η_0. Now, we can apply our results for NTK parametrization from above, and transfer them to standard parametrization by using $\theta_t^{\text{std}} = H^{\frac{1}{2}}\theta_t^{\text{ntk}}$, $f^{\text{std}}(x, \theta_t^{\text{std}}) = f^{\text{ntk}}(x, H^{-\frac{1}{2}}\theta_t^{\text{std}})$.

E.3 Redundancies when using the same learning rate for standard parametrization

In the previous section, we established the equivalence between training a neural network in NTK parametrization with learning rate η_0, and a neural network in standard parametrization with layer-dependent learning rate $\eta_0 H$. By definition, the learning rate for the first layer is $\frac{1}{n_0}\eta_0 = \frac{1}{d}\eta_0$, and the one for the biases is η_0. The learning rate for the other weight matrices is $\frac{1}{n_{l-1}}\eta_0 = \frac{1}{n}\eta_0$, for $l = 2, \ldots, L+1$. Note that the convergence of the learning rates to 0 for $n \to \infty$ is necessary to stabilize the gradient.

The learning rate that was used in the proof of Lee et al. (2019) is $\frac{1}{n}\eta_0$ for any layer. In the following, we will argue that this effectively leads to the first layer, and the biases not being trained in the infinite-width limit. For simplicity, let $\beta = 0$. Remember that Lee et al. (2019) shows that using the learning rate $\frac{1}{n}\eta_0$ for each

[11] The existence of a unique solution of this ODE will follow from the relation to the gradient flow under NTK parametrization.

layer in standard parametrization, leads to the trained network for large width being driven by the standard parametrization NTK

$$\frac{1}{n}J^{\text{std}}(x',\theta_0)J^{\text{std}}(x,\theta_0)^\top = \frac{1}{n}\sum_{l=1}^{L+1}\left(J^{\text{std}}(x',W_0^l)J^{\text{std}}(x,W_0^l)^\top + J^{\text{std}}(x',b_0^l)J^{\text{std}}(x,b_0^l)^\top\right). \tag{62}$$

By using the equivalences from the previous section, we may write for $l = 2,\dots,L+1$ (using $H_{w,l} = \frac{1}{n}I_{n_{l-1}n_l}$):

$$\frac{1}{n}J^{\text{std}}(x',W_0^l)J^{\text{std}}(x,W_0^l)^\top = \left(J^{\text{std}}(H_{w,l}^{\frac{1}{2}}\sqrt{n}W_0^l)H_{w,l}^{\frac{1}{2}}\right)\left(J^{\text{std}}(H_{w,l}^{\frac{1}{2}}\sqrt{n}W_0^l)H_{w,l}^{\frac{1}{2}}\right)^\top \tag{63}$$

$$= J^{\text{ntk}}(\sqrt{n}W_0^l)J^{\text{ntk}}(\sqrt{n}W_0^l)^\top. \tag{64}$$

This is equal to the empirical NTK under the NTK parametrization for the weights $\sqrt{n}W_{0,i,j}^l \sim \mathcal{N}(0,\sigma_{w,l})$ of the l-th layer. However, for the first layer, we get (using $H_{w,1} = \frac{1}{d}I_{dn}$)

$$\frac{1}{n}J^{\text{std}}(x',W_0^1)J^{\text{std}}(x,W_0^1)^\top = \frac{d}{n}\left(J^{\text{std}}(H_{w,1}^{\frac{1}{2}}\sqrt{d}W_0^1)H_{w,1}^{\frac{1}{2}}\right)\left(J^{\text{std}}(H_{w,1}^{\frac{1}{2}}\sqrt{d}W_0^1)H_{w,1}^{\frac{1}{2}}\right)^\top \tag{65}$$

$$= \frac{d}{n}J^{\text{ntk}}(\sqrt{d}W_0^1)J^{\text{ntk}}(\sqrt{d}W_0^1)^\top \tag{66}$$

$$\to 0, \text{ for } n\to\infty, \tag{67}$$

as $J^{\text{ntk}}(\sqrt{d}W_0^1)J^{\text{ntk}}(\sqrt{d}W_0^1)^\top$ converges by Theorem E.1. Similarly for the biases, for $l = 1,\dots,L+1$:

$$\frac{1}{n}J^{\text{std}}(x',b_0^l)J^{\text{std}}(x,b_0^l)^\top = \frac{1}{n}J^{\text{ntk}}(x',b_0^l)J^{\text{ntk}}(x,b_0^l)^\top \to 0, \text{ for } n\to\infty. \tag{68}$$

Thus, the analytical standard parametrization NTK of Lee et al. (2019) doesn't depend on the contribution of the gradient with respect to the first layer and the biases. In other words, using the learning rate $\frac{1}{n}\eta_0$ for the first layer and the biases leads to them not being trained for large widths.

Instead, one may scale the learning rates "correctly", as motivated by the NTK parametrization in the previous section. For large widths n, the trained network is then governed by the following modified NTK for standard parametrization:

$$J^{\text{std}}(\theta)HJ^{\text{std}}(\theta) = J^{\text{std}}(W^1)J^{\text{std}}(W^1)^\top + J^{\text{std}}(b^1)J^{\text{std}}(b^1)^\top \tag{69}$$

$$+\sum_{l=2}^{L+1}\left(\frac{1}{n}J^{\text{std}}(W^l)J^{\text{std}}(W^l)^\top + J^{\text{std}}(b^l)J^{\text{std}}(b^l)^\top\right). \tag{70}$$

F PROOF FOR REGULARIZED GRADIENT FLOW

F.1 Exponential Decay of the Regularized Gradient and Closeness of Parameters to their Initial Value

Lemma F.1. *Let $\beta \geq 0$. We have for any $t \geq 0$: $\|g(\theta_t)\|_2 \leq \|g(\theta_0)\|_2$. Further, for any $\delta_0 > 0$, there is $R_0 > 0$, such that for n large enough, with probability of at least $1-\delta_0$ over random initialization, $\|g(\theta_0)\|_2 \leq R_0$.*

Proof. Using the chain rule and the definition of the gradient flow, we have

$$\frac{d}{dt}\mathcal{L}^\beta(\theta_t) = \nabla_\theta\mathcal{L}^\beta(\theta_t)^\top\frac{d\theta_t}{dt} = -\eta_0\nabla_\theta\mathcal{L}^\beta(\theta_t)^\top\nabla_\theta\mathcal{L}^\beta(\theta_t) = -\eta_0\|\nabla_\theta\mathcal{L}^\beta(\theta_t)\|_2^2 \leq 0. \tag{71}$$

Thus,

$$\frac{1}{2}\|g(\theta_t)\|_2^2 \leq \frac{1}{2}\|g(\theta_t)\|_2^2 + \frac{1}{2}\beta\|\theta_t-\theta_0\|_2^2 = \mathcal{L}^\beta(\theta_t) \leq \mathcal{L}^\beta(\theta_0) = \frac{1}{2}\|g(\theta_0)\|_2^2. \tag{72}$$

Hence, $\|g(\theta_t)\|_2 \leq \|g(\theta_0)\|_2$. Further, note that $f(\theta_0)$ converges in distribution to a Gaussian with mean zero and covariance given by the NNGP kernel (Lee et al., 2018). Thus, for n large enough, one can bound $\|g(\theta_0)\|_2$ with high probability. □

The proof of this Lemma does not apply to the weight-decay regularizer $\frac{1}{2}\|\theta\|_2^2$ analyzed in Lewkowycz and Gur-Ari (2020): In general, $\frac{1}{2}\|\theta_0\|_2^2 \neq 0$, and thus we can not use $\mathcal{L}^\beta(\theta_0) = \frac{1}{2}\|g(\theta_0)\|_2^2$. They show that using gradient flow with this regularizer leads to $f_{\theta_\infty}(\cdot) = 0$ in the infinite-width limit, and we are thus not in the NTK regime.

We restate Theorem 4.2 for convenience.

Theorem 4.2. *Let $\beta \geq 0$. Let $\delta_0 > 0$ arbitrarily small. There are $K', K, R_0, c_\beta > 0$, such that for n large enough, the following holds with probability of at least $1-\delta_0$ over random initialization, when applying regularized gradient flow with learning rate $\eta = \eta_0$:*

$$\left\| \frac{d\theta_t}{dt} \right\|_2 = \eta_0 \|\nabla_\theta \mathcal{L}^\beta(\theta_t)\|_2 \leq \eta_0 K R_0 e^{-\eta_0 c_\beta t}, \tag{12}$$

$$\|\theta_t - \theta_0\|_2 \leq \frac{KR_0}{c_\beta}\left(1 - e^{-\eta_0 c_\beta t}\right) < \frac{KR_0}{c_\beta} =: C, \tag{13}$$

$$\forall \|x\|_2 \leq 1 : \|J(x,\theta_t) - J(x,\theta_0)\|_2 \leq \frac{(\log n)^c}{\sqrt{n}} K' C, \tag{14}$$

$$\|J(\theta_t) - J(\theta_0)\|_2 \leq \frac{(\log n)^c}{\sqrt{n}} K C. \tag{15}$$

Proof. Using Lemma F.1, there is $R_0 > 0$, such that for n large enough, with probability of at least $1 - \frac{1}{3}\delta_0$ over random initialization, $\|g(\theta_0)\|_2 \leq R_0$. Further, using Lemma A.1, let K be the constant for local Lipschitzness/Boundedness of the Jacobian with probability $1 - \frac{1}{3}\delta_0$ for n large enough. Finally, by Lemma E.2, for n large enough, with probability of at least $1 - \frac{1}{3}\delta_0$ over random initialization, the minimum eigenvalue of the empirical NTK is lower bounded: $\lambda_{\min}(J(\theta_0)J(\theta_0)^\top) \geq \frac{1}{2}\lambda_{\min}(\mathbf{\Theta})$.[12] For n large enough, these three events hold with probability of at least $1 - \delta_0$ over random initialization. In the following, we consider such initializations θ_0.

Define $c_\beta := \frac{1}{2}\beta$ for $\beta > 0$, and $c_\beta := \frac{1}{3}\lambda_{\min}(\mathbf{\Theta})$ for $\beta = 0$.[13] Let $C := \frac{KR_0}{c_\beta}$. By Lemma A.1, the gradient flow ODE has a unique solution as long as $\theta_t \in B(\theta_0, C)$. Consider $t_1 := \inf\{t \geq 0 : \|\theta_t - \theta_0\|_2 \geq C\}$. In the following, let $t \leq t_1$. Remember that

$$\frac{d\theta_t}{dt} = -\eta_0 \nabla_\theta \mathcal{L}^\beta(\theta_t) = -\eta_0 \left(J(\theta_t)^\top g(\theta_t) + \beta(\theta_t - \theta_0)\right). \tag{73}$$

We want to show that the gradient $\nabla_\theta \mathcal{L}^\beta(\theta_t)$ of the regularized loss converges to 0 quickly, and hence θ_t doesn't move much. For $\beta = 0$, its norm is $\|J(\theta_t)^\top g(\theta_t)\|_2$ and hence Lee et al. (2019) and related proofs showed that $\|g(\theta_t)\|_2$ converges to 0 quickly. However, for $\beta > 0$, this is not the case, as the training error won't converge to 0. Instead, we directly look at the dynamics of the norm of the gradient:

$$\frac{d}{dt}\|\nabla_\theta \mathcal{L}^\beta(\theta_t)\|_2^2 = 2\left(\nabla_\theta \mathcal{L}^\beta(\theta_t)\right)^\top \nabla_\theta^2 \mathcal{L}^\beta(\theta_t) \frac{d\theta_t}{dt} \tag{74}$$

$$= -2\eta_0 \left(\nabla_\theta \mathcal{L}^\beta(\theta_t)\right)^\top \nabla_\theta^2 \mathcal{L}^\beta(\theta_t) \left(\nabla_\theta \mathcal{L}^\beta(\theta_t)\right). \tag{75}$$

The Hessian $\nabla_\theta^2 \mathcal{L}^\beta(\theta_t) \in \mathbb{R}^{p \times p}$ of the regularized loss is given by

$$\nabla_\theta^2 \mathcal{L}^\beta(\theta_t) = g(\theta_t)^\top \nabla_\theta^2 f(\theta_t) + J(\theta)^\top J(\theta_t) + \beta I_p. \tag{76}$$

Here, $\nabla_\theta^2 f(\theta) \in \mathbb{R}^{N \times p \times p}$, and $g(\theta_t)^\top \nabla_\theta^2 f(\theta_t) = \sum_{i=1}^N g(\mathbf{x}_i, \theta_t) \nabla_\theta^2 f(\mathbf{x}_i, \theta_t)$. By using the triangle inequality and Cauchy-Schwarz,

$$\|g(\theta_t)^\top \nabla_\theta^2 f(\theta)\|_2 \leq \sum_{i=1}^N |g(\mathbf{x}_i, \theta_t)| \|\nabla_\theta^2 f(\mathbf{x}_i, \theta_t)\|_2 \leq \|g(\theta_t)\|_2 \sqrt{\sum_{i=1}^N \|\nabla_\theta^2 f(\mathbf{x}_i, \theta_t)\|_2^2}. \tag{77}$$

[12] We will only need this for $\beta = 0$.

[13] We note that we could choose any constant smaller than β for $\beta > 0$, and similarly, and constant smaller than $\lambda_{\min}(\mathbf{\Theta})$ for $\beta = 0$.

Using Lemma F.1, we have $\|g(\theta_t)\|_2 \leq \|g(\theta_0)\|_2 \leq R_0$. Further, by Lemma A.1, we have $\|\nabla^2_\theta f(\mathbf{x}_i, \theta_t)\|_2 \leq \frac{(\log n)^c}{\sqrt{n}} K'$. Thus (with $K = \sqrt{N}K'$),

$$\|g(\theta_t)^\top \nabla^2_\theta f(\theta)\|_2 \leq \frac{(\log n)^c}{\sqrt{n}} K R_0. \tag{78}$$

As $g(\theta_t)^\top \nabla^2_\theta f(\theta)$ is symmetric, it follows for its minimum eigenvalue, that

$$\lambda_{\min}\left(g(\theta_t)^\top \nabla^2_\theta f(\theta)\right) \geq -\|g(\theta_t)^\top \nabla^2_\theta f(\theta)\|_2 \geq -\frac{(\log n)^c}{\sqrt{n}} K R_0. \tag{79}$$

Now consider $\beta > 0$. Then, we can follow that for n large enough, the smallest eigenvalue of the Hessian of the regularized loss is positive:

$$\lambda_{\min}\left(\nabla_\theta \mathcal{L}^\beta(\theta_t)\right) \geq -\frac{(\log n)^c}{\sqrt{n}} K R_0 + 0 + \beta \geq \frac{\beta}{2} =: c_\beta. \tag{80}$$

Thus,

$$\frac{d}{dt}\|\nabla_\theta \mathcal{L}^\beta(\theta_t)\|_2^2 \leq -2\eta_0 c_\beta \|\nabla_\theta \mathcal{L}^\beta(\theta_t)\|_2^2. \tag{81}$$

In Remark F.2 we will show, how to modify the proof such that this step is still valid for $\beta = 0$.

By Gronwalls inequality, it follows (for $\beta \geq 0$), that

$$\|\nabla_\theta \mathcal{L}^\beta(\theta_t)\|_2^2 \leq e^{-2\eta_0 c_\beta t} \|\nabla_\theta \mathcal{L}^\beta(\theta_0)\|_2^2. \tag{82}$$

Thus,

$$\|\nabla_\theta \mathcal{L}^\beta(\theta_t)\|_2 \leq e^{-\eta_0 c_\beta t} \|\nabla_\theta \mathcal{L}^\beta(\theta_0)\|_2 \leq e^{-\eta_0 c_\beta t} \|J(\theta_0)^\top g(\theta_0)\|_2 \leq e^{-\eta_0 c_\beta t} \|J(\theta_0)\|_2 \|g(\theta_0)\|_2 \leq K R_0 e^{-\eta_0 c_\beta t}. \tag{83}$$

Hence, for the distance of parameters from initialization

$$\|\theta_t - \theta_0\|_2 = \left\| \int_0^t \frac{d\theta_u}{du} du \right\|_2 \leq \int_0^t \left\| \frac{d\theta_u}{du} \right\|_2 du \leq \eta_0 K R_0 \int_0^t e^{-\eta_0 c_\beta u} du = \frac{K R_0}{c_\beta}(1 - e^{-\eta_0 c_\beta t}). \tag{84}$$

Thus, for $t \leq t_1$, $\|\theta_t - \theta_0\|_2 < C$. By continuity, $t_1 = \infty$. Using local Lipschitzness, we can further bound the distance of the Jacobian at any $\|x\|_2 \leq 1$:

$$\|J(x, \theta_t) - J(x, \theta_0)\|_2 \leq \frac{(\log n)^c}{\sqrt{n}} K' \|\theta_t - \theta_0\|_2 \leq \frac{(\log n)^c}{\sqrt{n}} K' C. \tag{85}$$

This finishes the proof of Theorem 4.2. □

Remark F.2. For $\beta = 0$, the Hessian is

$$\nabla^2_\theta \mathcal{L}^0(\theta_t) = g(\theta_t)^\top \nabla^2_\theta f(\theta_t) + J(\theta_t)^\top J(\theta_t). \tag{86}$$

For $\beta > 0$, we just used that $J(\theta_t)^\top J(\theta_t)$ is positive semi-definite, as βI dominates the negative eigenvalues of the first term of the Hessian. For $\beta = 0$, this is not enough. $J(\theta_t)^\top J(\theta_t) \in \mathbb{R}^{p \times p}$ shouldn't be confused with the NTK $J(\theta_t)J(\theta_t)^\top \in \mathbb{R}^{N \times N}$. However, they share the have the same nonzero eigenvalues. For $p > N$ (which is the case for n large enough), $J(\theta_t)^\top J(\theta_t)$ will additionally have the eigenvalue 0 with multiplicity of at least $p - N$. Thus, we can't naively lower bound the minimum eigenvalue of the Hessian with the minimum eigenvalue of $J(\theta_t)^\top J(\theta_t)$.

Luckily, $\nabla_\theta \mathcal{L}^0(\theta_t) = J(\theta_t)^\top g(\theta_t)$ is in the row-span of $J(\theta_t)$. This is orthogonal to the nullspace of $J(\theta_t)$, i.e. the eigenspace corresponding to the eigenvalue 0 of $J(\theta_t)$. Thus, $\nabla_\theta \mathcal{L}^0(\theta_t)$ only "uses" the positive eigenvalues of $J(\theta_t)^\top J(\theta_t)$. The smallest positive eigenvalue of $J(\theta_t)^\top J(\theta_t)$ is equal to the smallest positive eigenvalue of the empirical NTK $J(\theta_t)J(\theta_t)^\top$, which is lower bounded by $\frac{1}{2}\lambda_{\min}(\mathbf{\Theta})$ on the high probability event we consider.

Hence, for n large enough,

$$\frac{d}{dt}\|\nabla_\theta \mathcal{L}^0(\theta_t)\|_2^2 = -2\eta_0 \left(\nabla_\theta \mathcal{L}^0(\theta_t)\right)^\top \nabla_\theta^2 \mathcal{L}^0(\theta_t) \left(\nabla_\theta \mathcal{L}^0(\theta_t)\right) \tag{87}$$

$$\leq -2\eta_0 \left(\lambda_{\min}\left(g(\theta_t)^\top \nabla_\theta^2 f(\theta)\right) + \frac{1}{2}\lambda_{\min}(\mathbf{\Theta})\right) \|\nabla_\theta \mathcal{L}^0(\theta_t)\|_2^2 \tag{88}$$

$$\leq -2\eta_0 \left(-\frac{(\log n)^c}{\sqrt{n}} K R_0 + \frac{1}{2}\lambda_{\min}(\mathbf{\Theta})\right) \|\nabla_\theta \mathcal{L}^0(\theta_t)\|_2^2 \tag{89}$$

$$\leq -2\eta_0 \frac{1}{3}\lambda_{\min}(\mathbf{\Theta}) \|\nabla_\theta \mathcal{L}^0(\theta_t)\|_2^2. \tag{90}$$

Defining $c_0 := \frac{1}{3}\lambda_{\min}(\mathbf{\Theta})$ makes it possible to continue with the proof above for $\beta \geq 0$. This is an alternative proof to Lee et al. (2019). It shows that in the unregularized case, it is important that the gradient flow lies in the row space of the Jacobian.

F.2 Closeness to the Linearized Network along the Regularized Gradient Flow

The goal of this section is to prove that the neural network along the regularized gradient flow stays close to the linearized network along the linear regularized gradient flow. Let us restate the following Theorem.

Theorem 4.3. *Let $\beta \geq 0$. Let $\delta_0 > 0$ be arbitrarily small. Then, there are C_1, C_2, such that for n large enough, with probability of at least $1 - \delta_0$ over random initialization,*

$$\sup_{t \geq 0} \|\theta_t - \theta_t^{\text{lin}}\|_2 \leq C_1 \frac{(\log n)^c}{\sqrt{n}}, \tag{16}$$

$$\forall \|x\|_2 \leq 1 : \sup_{t \geq 0} \|f(x, \theta_t) - f_{\theta_0}^{\text{lin}}(x, \theta_t^{\text{lin}})\|_2 \leq C_2 \frac{(\log n)^c}{\sqrt{n}}. \tag{17}$$

Proof. The proof of Lee et al. (2019) used that training error converges to 0. Thus, we have to use a different approach, which also provides a more straightforward and intuitive proof for $\beta = 0$. Remember that

$$f^{\text{lin}}(x, \theta) = f(x, \theta_0) + J(x, \theta_0)(\theta - \theta_0), \text{ and } \frac{d\theta^{\text{lin}}}{dt} = -\eta_0 \left(J(\theta_0)^\top g^{\text{lin}}(\theta_t^{\text{lin}}) + \beta(\theta_t^{\text{lin}} - \theta_0)\right). \tag{91}$$

To prove the second part of the Theorem, we will use

$$\|f(x, \theta_t) - f^{\text{lin}}(x, \theta_t^{\text{lin}})\|_2 \leq \|f(x, \theta_t) - f^{\text{lin}}(x, \theta_t)\|_2 + \|f^{\text{lin}}(x, \theta_t) - f^{\text{lin}}(x, \theta_t^{\text{lin}})\|_2. \tag{92}$$

We will start by bounding the first term. Next, we will bound $\|\theta_t - \theta_t^{\text{lin}}\|_2$, and use this to bound the second term.

First step: To bound $\|f(x, \theta_t) - f^{\text{lin}}(x, \theta_t)\|_2$, we compute

$$\left\| \frac{d}{dt} \left(f(x, \theta_t) - f^{\text{lin}}(x, \theta_t)\right) \right\|_2 = \left\| \left(J(x, \theta_t) - J(x, \theta_0)\right) \frac{d\theta_t}{dt} \right\|_2 \tag{93}$$

$$\leq \|J(x, \theta_t) - J(x, \theta_0)\|_2 \left\| \frac{d\theta_t}{dt} \right\|_2 \tag{94}$$

$$\leq \frac{(\log n)^c}{\sqrt{n}} K' C \eta_0 K R_0 e^{-\eta_0 c_\beta t}, \tag{95}$$

where we used Theorem 4.2 in the last step. Now, we can bound

$$\|f(x, \theta_t) - f^{\text{lin}}(x, \theta_t)\|_2 \leq \frac{(\log n)^c}{\sqrt{n}} K' C \eta_0 K R_0 \int_0^t e^{-\eta_0 c_\beta u} du \leq \frac{(\log n)^c}{\sqrt{n}} K' C \frac{K R_0}{c_\beta} = \frac{(\log n)^c}{\sqrt{n}} K' C^2. \tag{96}$$

In particular, for the difference at the training points, $\|f(\theta_t) - f^{\text{lin}}(\theta_t)\|_2 \leq \frac{(\log n)^c}{\sqrt{n}} K C^2$.

Second step: Now, we will bound the difference between $\theta_t - \theta_t^{\text{lin}}$. We can write

$$\frac{d\theta_t}{dt} = -\eta_0 \left(J(\theta_t)^\top g(\theta_t) + \beta(\theta_t - \theta_0)\right) \tag{97}$$

$$= -\eta_0 \left(\left(J(\theta_t) - J(\theta_0)\right)^\top g(\theta_t) + J(\theta_0)^\top \left(g(\theta_t) - g^{\text{lin}}(\theta_t)\right) + J(\theta_0)^\top g^{\text{lin}}(\theta_t) + \beta(\theta_t - \theta_0)\right) \tag{98}$$

$$= -\eta_0 \Delta_t - \eta_0 \left(J(\theta_0)^\top g^{\text{lin}}(\theta_t) + \beta(\theta_t - \theta_0)\right), \tag{99}$$

where we define $\Delta_t := \left(J(\theta_t) - J(\theta_0)\right)^\top g(\theta_t) + J(\theta_0)^\top \left(g(\theta_t) - g^{\text{lin}}(\theta_t)\right)$. We will now bound $\|\Delta_t\|_2$. For the first term, using Theorem 4.2:

$$\|\left(J(\theta_t) - J(\theta_0)\right)^\top g(\theta_t)\|_2 \leq \|J(\theta_t) - J(\theta_0)\|_2 \|g(\theta_t)\|_2 \leq \frac{(\log n)^c}{\sqrt{n}} KCR_0. \tag{100}$$

For the second term, using Theorem 4.2 and the bound we derived in the first step,

$$\|J(\theta_0)^\top \left(g(\theta_t) - g^{\text{lin}}(\theta_t)\right)\|_2 = \|J(\theta_0)^\top \left(f(\theta_t) - f^{\text{lin}}(\theta_t)\right)\|_2 \leq \|J(\theta_0)\|_2 \|f(\theta_t) - f^{\text{lin}}(\theta_t)\|_2 \leq \frac{(\log n)^c}{\sqrt{n}} K^2 C^2. \tag{101}$$

Thus, defining $K^\Delta := KCR_0 + K^2C^2$, we can bound $\|\Delta_t\|_2 \leq \frac{(\log n)^c}{\sqrt{n}} K^\Delta$. Now, we can compute

$$\frac{d}{dt}(\theta_t - \theta_t^{\text{lin}}) = -\eta_0 \Delta_t - \eta_0 \left(J(\theta_0)^\top g^{\text{lin}}(\theta_t) + \beta(\theta_t - \theta_0)\right) + \eta_0 \left(J(\theta_0)^\top g^{\text{lin}}(\theta_t^{\text{lin}}) + \beta(\theta_t^{\text{lin}} - \theta_0)\right) \tag{102}$$

$$= -\eta_0 \Delta_t - \eta_0 \left(J(\theta_0)^\top \left(g^{\text{lin}}(\theta_t) - g^{\text{lin}}(\theta_t^{\text{lin}})\right) + \beta(\theta_t - \theta_t^{\text{lin}})\right) \tag{103}$$

$$= -\eta_0 \Delta_t - \eta_0 \left(J(\theta_0)^\top J(\theta_0)(\theta_t - \theta_t^{\text{lin}}) + \beta(\theta_t - \theta_t^{\text{lin}})\right) \tag{104}$$

$$= -\eta_0 \Delta_t - \eta_0 \left(J(\theta_0) J(\theta_0)^\top + \beta I\right)(\theta_t - \theta_t^{\text{lin}}). \tag{105}$$

By treating this as an inhomogeneous linear ODE in $\theta_t - \theta_t^{\text{lin}}$, we get

$$\theta_t - \theta_t^{\text{lin}} = \int_0^t e^{-\eta_0 \left(J(\theta_0) J(\theta_0)^\top + \beta I\right)(t-u)} (-\eta_0 \Delta_u) du. \tag{106}$$

Hence (using $\|e^{-A}\|_2 \leq e^{-\lambda_{\min}(A)}$),

$$\|\theta_t - \theta_t^{\text{lin}}\|_2 \leq \int_0^t \|e^{-\eta_0 \left(J(\theta_0) J(\theta_0)^\top + \beta I\right)(t-u)}\|_2 \eta_0 \|\Delta_u\|_2 du \tag{107}$$

$$\leq \int_0^t e^{-\eta_0 (c_0 + \beta)(t-u)} \eta_0 \frac{(\log n)^c}{\sqrt{n}} K^\Delta du \tag{108}$$

$$\leq \frac{(\log n)^c}{\sqrt{n}} \frac{K^\Delta}{c_0 + \beta}. \tag{109}$$

Thus, $\sup_t \|\theta_t - \theta_t^{\text{lin}}\|_2 \leq \frac{K^\Delta}{c_0 + \beta} \frac{(\log n)^c}{\sqrt{n}}$.

Third step: Using the bound on $\|\theta_t - \theta_t^{\text{lin}}\|_2$, we can easily bound $\|f^{\text{lin}}(x, \theta_t) - f^{\text{lin}}(x, \theta_t^{\text{lin}})\|_2$:

$$\|f^{\text{lin}}(x, \theta_t) - f^{\text{lin}}(x, \theta_t^{\text{lin}})\|_2 = \|J(x, \theta_0)(\theta_t - \theta_t^{\text{lin}})\|_2 \leq \|J(x, \theta_0)\|_2 \|\theta_t - \theta_t^{\text{lin}}\|_2 \leq K' \frac{K^\Delta}{c_0 + \beta} \frac{(\log n)^c}{\sqrt{n}}. \tag{110}$$

By using equation (92), we can now finish the proof:

$$\|f(x, \theta_t) - f^{\text{lin}}(x, \theta_t^{\text{lin}})\|_2 \leq \left(K' C^2 + K' \frac{K^\Delta}{c_0 + \beta}\right) \frac{(\log n)^c}{\sqrt{n}}. \tag{111}$$

□

G PROOF FOR REGULARIZED GRADIENT DESCENT

G.1 Geometric Decay of the regularized gradient and closeness of parameters to their initial value

Theorem G.1. *Let $\beta \geq 0$. Let $\delta_0 > 0$ arbitrarily small. There are $K', K, R_0, c_\beta, \eta_{\max} > 0$, such that for n large enough, the following holds with probability of at least $1 - \delta_0$ over random initialization, when applying regularized gradient descent with learning rate $\eta = \eta_0 \leq \eta_{\max}$:*

$$\|\theta_{t+1} - \theta_t\|_2 = \eta_0 \|\nabla_\theta \mathcal{L}^\beta(\theta_t)\|_2 \leq \eta_0 K R_0 \left(1 - \eta_0 c_\beta\right)^t, \tag{112}$$

$$\|\theta_t - \theta_0\|_2 < \frac{K R_0}{c_\beta} =: C, \tag{113}$$

$$\forall \|x\|_2 \leq 1 : \|J(x, \theta_t) - J(x, \theta_0)\|_2 \leq \frac{(\log n)^c}{\sqrt{n}} K' C, \tag{114}$$

$$\|J(\theta_t) - J(\theta_0)\|_2 \leq \frac{(\log n)^c}{\sqrt{n}} K C. \tag{115}$$

Proof. We consider the same high probability event as in the proof for the regularized gradient flow. Define $c_\beta := \frac{1}{2}\beta$ for $\beta > 0$, and $c_\beta := \frac{1}{3}\lambda_{\min}(\mathbf{\Theta})$ for $\beta = 0$. Let $C := \frac{K R_0}{c_\beta}$.

We will prove the first two inequalities by induction. For $t = 0$:

$$\|\nabla_\theta \mathcal{L}^\beta(\theta_t)\|_2 = \|J(\theta_0)^\top g(\theta_0)\|_2 \leq K R_0. \tag{116}$$

Now, assume it holds true for $s \leq t$. We want to bound $\|\theta_{t+1} - \theta_t\|_2 = \eta_0 \|\nabla_\theta \mathcal{L}^\beta(\theta_t)\|_2$. Remember that $\nabla_\theta \mathcal{L}^\beta(\theta_t) = J(\theta_t)^\top g(\theta_t) + \beta(\theta_t - \theta_0)$. We write

$$\begin{aligned}
\|\nabla_\theta \mathcal{L}^\beta(\theta_t)\|_2^2 =& \|\nabla_\theta \mathcal{L}^\beta(\theta_{t-1}) + \nabla_\theta \mathcal{L}^\beta(\theta_t) - \nabla_\theta \mathcal{L}^\beta(\theta_{t-1})\|_2^2 && (117)\\
=& \|\nabla_\theta \mathcal{L}^\beta(\theta_{t-1})\|_2^2 && (118)\\
&+ 2\nabla_\theta \mathcal{L}^\beta(\theta_{t-1})^\top \left(\nabla_\theta \mathcal{L}^\beta(\theta_t) - \nabla_\theta \mathcal{L}^\beta(\theta_{t-1})\right) && (119)\\
&+ \|\nabla_\theta \mathcal{L}^\beta(\theta_t) - \nabla_\theta \mathcal{L}^\beta(\theta_{t-1})\|_2^2. && (120)
\end{aligned}$$

In the following, we will look at how to bound the second and the third term. We have:

$$\begin{aligned}
\nabla_\theta \mathcal{L}^\beta(\theta_t) - \nabla_\theta \mathcal{L}^\beta(\theta_{t-1}) &= \nabla_\theta \mathcal{L}^\beta(\theta_{t-1} - \eta_0 \nabla_\theta \mathcal{L}^\beta(\theta_{t-1})) - \nabla_\theta \mathcal{L}^\beta(\theta_{t-1}) && (121)\\
&= -\int_0^{\eta_0} \left(\nabla_\theta^2 \mathcal{L}^\beta\left(\theta_{t-1} - u \nabla_\theta \mathcal{L}^\beta(\theta_{t-1})\right)\right) \cdot \nabla_\theta \mathcal{L}^\beta(\theta_{t-1}) du. && (122)
\end{aligned}$$

As in the proof for the gradient flow, the following part only holds for $\beta > 0$. Note, that for any $u \in [0, \eta_0]$, $\theta_{t-1} - u \nabla_\theta \mathcal{L}^\beta(\theta_{t-1}) \in B(\theta_0, C)$, as we know by induction that $\theta_{t-1}, \theta_t \in B(\theta_0, C)$. Thus, similar to the proof for the gradient flow, for n large enough, $\forall u \in [0, \eta_0]$:

$$\lambda_{\min}\left(\nabla_\theta^2 \mathcal{L}^\beta\left(\theta_{t-1} - u \nabla_\theta \mathcal{L}^\beta(\theta_{t-1})\right)\right) \geq \frac{3}{2} c_\beta. \tag{123}$$

Note that we are using $\frac{3}{2} c_\beta = \frac{3}{4}\beta$, which is slightly higher than c_β which we used in the gradient flow case, to arrive at the equivalent result in the end. For the second term (119) we get,

$$\begin{aligned}
& 2\nabla_\theta \mathcal{L}^\beta(\theta_{t-1})^\top \left(\nabla_\theta \mathcal{L}^\beta(\theta_t) - \nabla_\theta \mathcal{L}^\beta(\theta_{t-1})\right) && (124)\\
=& -2 \int_0^{\eta_0} \nabla_\theta \mathcal{L}^\beta(\theta_{t-1}) \left(\nabla_\theta^2 \mathcal{L}^\beta\left(\theta_{t-1} - u \nabla_\theta \mathcal{L}^\beta(\theta_{t-1})\right)\right) \cdot \nabla_\theta \mathcal{L}^\beta(\theta_{t-1}) du && (125)\\
\leq& -2\eta_0 \frac{3}{2} c_\beta \|\nabla_\theta \mathcal{L}^\beta(\theta_{t-1})\|_2^2. && (126)
\end{aligned}$$

Further, we have for any $\theta \in B(\theta_0, C)$:

$$\|\nabla^2_\theta \mathcal{L}^\beta(\theta)\|_2 \leq \|g(\theta)^\top \nabla^2_\theta f(\theta)\|_2 + \|J(\theta)^\top J(\theta)\|_2 + \beta I_p \tag{127}$$

$$\leq \frac{(\log n)^c}{\sqrt{n}} K R_0 + \lambda_{\max}\left(J(\theta)^\top J(\theta)\right) + \beta \tag{128}$$

$$\leq \frac{(\log n)^c}{\sqrt{n}} K R_0 + 2\lambda_{\max}(\mathbf{\Theta}) + \beta \tag{129}$$

$$\leq 2(\lambda_{\max}(\mathbf{\Theta}) + \beta), \tag{130}$$

for n large enough. Using this with $\theta = \theta_{t-1} - u\nabla_\theta \mathcal{L}^\beta(\theta_{t-1})$, we get for the third term (120),

$$\|\nabla_\theta \mathcal{L}^\beta(\theta_t) - \nabla_\theta \mathcal{L}^\beta(\theta_{t-1})\|_2^2 = \left\| \int_0^{\eta_0} \left(\nabla^2_\theta \mathcal{L}^\beta\left(\theta_{t-1} - u\nabla_\theta \mathcal{L}^\beta(\theta_{t-1})\right)\right) \cdot \nabla_\theta \mathcal{L}^\beta(\theta_{t-1}) du \right\|_2^2 \tag{131}$$

$$\leq \left(\int_0^{\eta_0} \|\nabla^2_\theta \mathcal{L}^\beta\left(\theta_{t-1} - u\nabla_\theta \mathcal{L}^\beta(\theta_{t-1})\right)\|_2 \|\nabla_\theta \mathcal{L}^\beta(\theta_{t-1})\|_2 du \right)^2 \tag{132}$$

$$\leq \eta_0^2 \left(2(\lambda_{\max}(\mathbf{\Theta}) + \beta)\right)^2 \|\nabla_\theta \mathcal{L}^\beta(\theta_{t-1})\|_2^2 \tag{133}$$

$$\leq \eta_0 c_\beta \|\nabla_\theta \mathcal{L}^\beta(\theta_{t-1})\|_2^2. \tag{134}$$

In the last inequality, we chose the learning rate $\eta_0 \leq \frac{c_\beta}{4(\lambda_{\max}(\mathbf{\Theta})+\beta)^2}$ small enough. Similarly to the gradient flow case, one can derive such bounds for $\beta = 0$. Summing up the three terms,

$$\|\nabla_\theta \mathcal{L}^\beta(\theta_t)\|_2^2 \leq (1 - 2\eta_0 \frac{3}{2} c_\beta + \eta_0 c_\beta)\|\nabla_\theta \mathcal{L}^\beta(\theta_{t-1})\|_2^2 = (1 - 2\eta_0 c_\beta)\|\nabla_\theta \mathcal{L}^\beta(\theta_{t-1})\|_2^2. \tag{135}$$

Thus, by Bernoulli's inequality and the induction hypothesis,

$$\|\nabla_\theta \mathcal{L}^\beta(\theta_t)\|_2 \leq \sqrt{1 - 2\eta_0 c_\beta}\|\nabla_\theta \mathcal{L}^\beta(\theta_{t-1})\|_2 \leq (1 - \eta_0 c_\beta)\|\nabla_\theta \mathcal{L}^\beta(\theta_{t-1})\|_2 \leq K R_0 (1 - \eta_0 c_\beta)^t. \tag{136}$$

Hence, $\|\theta_{t+1} - \theta_t\|_2 = \eta_0 \|\nabla_\theta \mathcal{L}^\beta(\theta_t)\|_2 \leq \eta_0 K R_0 (1 - \eta_0 c_\beta)^t$. From this, we can follow that

$$\|\theta_{t+1} - \theta_0\|_2 \leq \sum_{u=0}^{t} \|\theta_{u+1} - \theta_u\|_2 \leq \eta_0 K R_0 \sum_{u=0}^{t} (1 - \eta_0 c_\beta)^u = \eta_0 K R_0 \frac{1 - (1 - \eta_0 c_\beta)^{t+1}}{\eta_0 c_\beta} < \frac{K R_0}{c_\beta} = C. \tag{137}$$

This proves the first two inequalities. The rest follows directly from the local Lipschitzness of the Jacobian, like in the proof for the gradient flow. □

G.2 Closeness to the linearized network along the regularized gradient descent

The following Theorem reads the same as in the gradient flow case, and the proof is very similar, which is why we only provide the main idea.

Theorem G.2. *Let $\beta \geq 0$. Let $\delta_0 > 0$ be arbitrarily small. Then, there are $C_1, C_2 > 0$, such that for n large enough, with probability of at least $1 - \delta_0$ over random initialization,*

$$\sup_{t \geq 0} \|\theta_t^{\text{lin}} - \theta_t\|_2 \leq C_1 \frac{(\log n)^c}{\sqrt{n}}, \quad \forall \|x\|_2 \leq 1 : \sup_{t \geq 0} \|f^{\text{lin}}(x, \theta_t^{\text{lin}}) - f(x, \theta_t)\|_2 \leq C_2 \frac{(\log n)^c}{\sqrt{n}}. \tag{138}$$

Proof Sketch. Remember that

$$f^{\text{lin}}(x, \theta) = f(x, \theta_0) + J(x, \theta_0)(\theta - \theta_0), \text{ and } \theta_{t+1}^{\text{lin}} = \theta_t^{\text{lin}} - \eta_0 \left(J(\theta_0)^\top g^{\text{lin}}(\theta_t^{\text{lin}}) + \beta(\theta_t^{\text{lin}} - \theta_0)\right). \tag{139}$$

The structure of the proof is the same as for the gradient flow. We will only show how to bound the term $\|f(x, \theta_t) - f^{\text{lin}}(x, \theta_t)\|_2$. The bounds for the other terms can be done similarly. In particular, we will show by induction that

$$\|f(x, \theta_t) - f^{\text{lin}}(x, \theta_t)\|_2 \leq \eta_0 \frac{(\log n)^c}{\sqrt{n}} K' C K R_0 \sum_{u=0}^{t-1} (1 - \eta_0 c_\beta)^u. \tag{140}$$

For $t = 0$, this is true. Now, assume this holds for $s \leq t$, then

$$\|f(x,\theta_{t+1}) - f^{\text{lin}}(x,\theta_{t+1})\|_2 \tag{141}$$

$$\leq \|f(x,\theta_t) - f^{\text{lin}}(x,\theta_t)\|_2 + \|f(x,\theta_{t+1}) - f(x,\theta_t) - \left(f^{\text{lin}}(x,\theta_{t+1}) - f^{\text{lin}}(x,\theta_t)\right)\|_2. \tag{142}$$

By the chain rule and the fundamental theorem of calculus, we can write

$$\|f(x,\theta_{t+1}) - f(x,\theta_t) - \left(f^{\text{lin}}(x,\theta_{t+1}) - f^{\text{lin}}(x,\theta_t)\right)\|_2 \tag{143}$$

$$= \left\| \int_0^{\eta_0} J\left(x, \theta_t - u\nabla_\theta \mathcal{L}^\beta(\theta_t)\right) \nabla_\theta \mathcal{L}^\beta(\theta_t) du - \int_0^{\eta_0} J(x,\theta_0)\nabla_\theta \mathcal{L}^\beta(\theta_t) du \right\|_2 \tag{144}$$

$$\leq \int_0^{\eta_0} \|J\left(x, \theta_t - u\nabla_\theta \mathcal{L}^\beta(\theta_t)\right) - J(x,\theta_0)\|_2 \|\nabla_\theta \mathcal{L}^\beta(\theta_t)\|_2 du \tag{145}$$

$$\leq \eta_0 \frac{(\log n)^c}{\sqrt{n}} K' C K R_0 \left(1 - \eta_0 c_\beta\right)^t. \tag{146}$$

In the last step we used Theorem G.1. Thus,

$$\|f(x,\theta_{t+1}) - f^{\text{lin}}(x,\theta_{t+1})\|_2 \leq \eta_0 \frac{(\log n)^c}{\sqrt{n}} K' C K R_0 \left(\sum_{u=0}^{t-1} \left(1 - \eta_0 c_\beta\right)^u + \left(1 - \eta_0 c_\beta\right)^t \right). \tag{147}$$

This finishes the induction proof. We can now further follow by using the geometric series that

$$\|f(x,\theta_t) - f^{\text{lin}}(x,\theta_t)\|_2 \leq \eta_0 \frac{(\log n)^c}{\sqrt{n}} K' C K R_0 \sum_{u=0}^{t-1} \left(1 - \eta_0 c_\beta\right)^u \tag{148}$$

$$< \eta_0 \frac{(\log n)^c}{\sqrt{n}} K' C K R_0 \frac{1}{\eta_0 c_\beta} \tag{149}$$

$$= \frac{(\log n)^c}{\sqrt{n}} K' C^2. \tag{150}$$

For the other inequalities one can proceed in the same manner, using the fundamental theorem of calculus and the geometric series. □

H SHIFTING THE NETWORK AT INITIALIZATION

Here, we prove that shifting the network at initialization makes it possible to include any prior mean, and compute the posterior mean with a single training run.

Theorem 5.1. *(Shifted Network.) Consider any function m. Given a random initialization θ_0, define shifted predictions $\tilde{f}_{\theta_0}(\mathbf{x},\theta)$ as follows:*

$$\tilde{f}_{\theta_0}(\mathbf{x},\theta) := f(\mathbf{x},\theta) - f(\mathbf{x},\theta_0) + m(\mathbf{x}). \tag{18}$$

Training this modified network (starting with θ_0) leads to the following output (in the infinite-width limit)

$$\tilde{f}_{\theta_0}(\mathbf{x}',\theta_\infty) = m(\mathbf{x}') + \mathbf{\Theta}_{\mathbf{x}',\mathbf{x}}(\mathbf{\Theta}_{\mathbf{x},\mathbf{x}} + \beta I)^{-1}(\mathbf{y} - m(\mathbf{x})). \tag{19}$$

This can be interpreted as the posterior mean of an NTK-GP with prior mean function m.

Proof. The Jacobian of the shifted network is equal to the Jacobian of the original network:

$$J_{\tilde{f}}(x,\theta) = J_f(x,\theta). \tag{151}$$

Define the shifted labels $\tilde{\mathbf{y}} := \mathbf{y} + f(\mathbf{x},\theta_0) - m(\mathbf{x})$. Then, $\tilde{f}(\mathbf{x},\theta) - \mathbf{y} = f(\mathbf{x},\theta) - \tilde{\mathbf{y}}$. Thus, training the network $\tilde{f}$ with regularized gradient flow/descent is equivalent to training f using the shifted labels $\tilde{\mathbf{y}}$, in the sense that the parameter update rule is the same. The latter leads to parameters θ_∞, for which (in the infinite-width limit)

$$f(\mathbf{x}',\theta_\infty) = f(\mathbf{x}',\theta_0) + \mathbf{\Theta}_{\mathbf{x}',\mathbf{x}}\left(\mathbf{\Theta}_{\mathbf{x},\mathbf{x}} + \beta I\right)^{-1}(\tilde{\mathbf{y}} - f(\mathbf{x},\theta_0)). \tag{152}$$

By adding $-f(\mathbf{x},\theta_0) + m(\mathbf{x})$ to both sides of the equation, and using $\tilde{\mathbf{y}} - f(\mathbf{x},\theta_0) = \mathbf{y} - m(\mathbf{x})$, we get

$$\tilde{f}(\mathbf{x}',\theta_\infty) = m(\mathbf{x}') + \mathbf{\Theta}_{\mathbf{x}',\mathbf{x}}\left(\mathbf{\Theta}_{\mathbf{x},\mathbf{x}} + \beta I\right)^{-1}(\mathbf{y} - m(\mathbf{x})). \tag{153}$$

□

I THE OUTPUT OF THE LINEARIZED NETWORK IS GAUSSIAN OVER RANDOM INITIALIZATIONS

Corollary I.1 (Convergence under Regularized Gradient Flow/Descent)**.** *Under regularized gradient flow/descent training, the output of a wide neural network converges in distribution to a Gaussian over random initialization as the width $n \to \infty$. Specifically, for test inputs $\mathbf{x}'$ and $t \to \infty$, the mean and covariance of the output distribution at convergence are*

$$\boldsymbol{\mu}(\mathbf{x}') = \boldsymbol{\Theta}_{\mathbf{x}',\mathbf{x}} \left(\boldsymbol{\Theta}_{\mathbf{x},\mathbf{x}} + \beta I\right)^{-1} \mathbf{y}, \tag{154}$$

$$\boldsymbol{\Sigma}(\mathbf{x}') = \mathbf{K}_{\mathbf{x}',\mathbf{x}'} + \boldsymbol{\Theta}_{\mathbf{x}',\mathbf{x}} \left(\boldsymbol{\Theta}_{\mathbf{x},\mathbf{x}} + \beta I\right)^{-1} \mathbf{K}_{\mathbf{x},\mathbf{x}} \left(\boldsymbol{\Theta}_{\mathbf{x},\mathbf{x}} + \beta I\right)^{-1} \boldsymbol{\Theta}_{\mathbf{x},\mathbf{x}'} \tag{155}$$

$$- \boldsymbol{\Theta}_{\mathbf{x}',\mathbf{x}} \left(\boldsymbol{\Theta}_{\mathbf{x},\mathbf{x}} + \beta I\right)^{-1} \mathbf{K}_{\mathbf{x},\mathbf{x}'} - \mathbf{K}_{\mathbf{x}',\mathbf{x}} \left(\boldsymbol{\Theta}_{\mathbf{x},\mathbf{x}} + \beta I\right)^{-1} \boldsymbol{\Theta}_{\mathbf{x},\mathbf{x}'}. \tag{156}$$

Note that the resulting covariance combines contributions from the NTK and NNGP kernels and therefore does not directly correspond to the posterior covariance of any GP.

Proof. As we showed, for large enough layer width,

$$f(\mathbf{x}', \theta_\infty) = f(\mathbf{x}', \theta_0) + \boldsymbol{\Theta}_{\mathbf{x}',\mathbf{x}} \left(\boldsymbol{\Theta}_{\mathbf{x},\mathbf{x}} + \beta I\right)^{-1} \left(\mathbf{y} - f(\mathbf{x}, \theta_0)\right). \tag{157}$$

$f(\mathbf{x}', \theta_0)$ and $f(\mathbf{x}, \theta_0)$ jointly converge to a Gaussian with mean zero and covariance matrix given through the NNGP-kernel $\mathbf{K}$. From this, it directly follows that $f(\mathbf{x}', \theta_\infty)$ converges to a Gaussian with the given mean and covariance matrices[14]. □

[14]The covariance matrix of $X + AY$, where X and Y are jointly Gaussian, is given by $\Sigma_X + A\Sigma_Y A^\top + A\Sigma_{X,Y} + \Sigma_{Y,X} A^\top$.