Global Convergence in Neural ODEs: Impact of Activation Functions

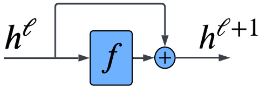
Tianxiang Gao[1], Siyuan Sun[2], Hailiang Liu[2], Hongyang Gao[2]

[1]DePaul University
[2]Iowa State University

ICLR 2025, April 24, 2025

From ResNet to Neural ODEs

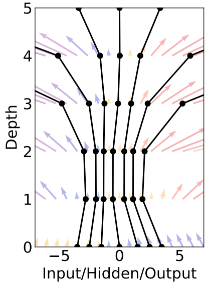
Residual Networks (ResNet):



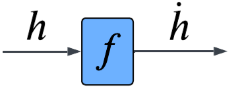
- Residual learning paradigm:

$$\boldsymbol{h}_{\ell+1} = \boldsymbol{h}_\ell + \boldsymbol{f}(\boldsymbol{h}_\ell, \boldsymbol{\theta}_\ell)$$

- A discrete sequence of transformations:



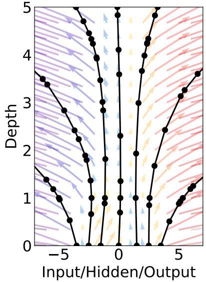
Neural Ordinary Differential Equations (ODEs):



- Infinitely many layers and infinitesimal scaling:

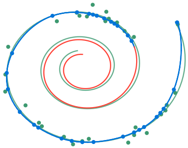
$$\dot{\boldsymbol{h}} = \boldsymbol{f}(\boldsymbol{h}_t, \boldsymbol{\theta}),$$

- A vector field continuously refining the state

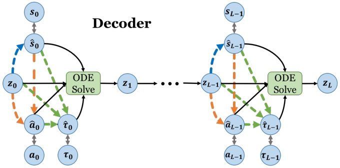


Applications of Neural ODEs

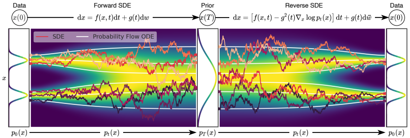
Thanks to their **continuous nature** and **parameter-sharing** efficiency, Neural ODEs have achieved success in a range of applications.



Time Series



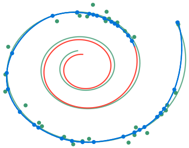
Reinforcement Learning



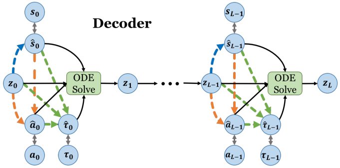
Score-based Generative Models

Applications of Neural ODEs

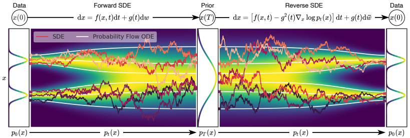
Thanks to their **continuous nature** and **parameter-sharing** efficiency, Neural ODEs have achieved success in a range of applications.



Time Series



Reinforcement Learning



Score-based Generative Models

Key Question

As a **continuous** model with **shared** parameters, under what conditions does a Neural ODE **converge** under a gradient-based method?

Setup

- We study a Neural ODE $f_{\boldsymbol{\theta}}(\boldsymbol{x})$ defined by the **forward ODE** with hidden state $\boldsymbol{h}_t$:

$$\dot{\boldsymbol{h}}_t = \boldsymbol{W}\phi(\boldsymbol{h}_t), \quad \forall t \in [0, T] \tag{1}$$

where $\boldsymbol{W} \in \mathbb{R}^{n \times n}$ and $\boldsymbol{W}_{ij} \overset{\text{iid}}{\sim} \mathcal{N}(0, 1/n)$.

Setup

- We study a Neural ODE $f_{\boldsymbol{\theta}}(\boldsymbol{x})$ defined by the **forward ODE** with hidden state $\boldsymbol{h}_t$:

$$\dot{\boldsymbol{h}}_t = \boldsymbol{W}\phi(\boldsymbol{h}_t), \quad \forall t \in [0, T] \tag{1}$$

  where $\boldsymbol{W} \in \mathbb{R}^{n\times n}$ and $\boldsymbol{W}_{ij} \overset{\text{iid}}{\sim} \mathcal{N}(0, 1/n)$.
- Given training data $\{(\boldsymbol{x}_i, y_i)\}_{i=1}^N$, we minimize the squared loss:

$$L(\boldsymbol{\theta}) = \sum_{i=1}^{N} \frac{1}{2}\left[f_{\boldsymbol{\theta}}(\boldsymbol{x}_i) - y_i\right]^2 = \frac{1}{2}\|\boldsymbol{u} - \boldsymbol{y}\|^2, \tag{2}$$

  where $\boldsymbol{u}$ is the prediction vector with $\boldsymbol{u}_i = f_\theta(\boldsymbol{x}_i)$.

Setup

- We study a Neural ODE $f_{\boldsymbol{\theta}}(\boldsymbol{x})$ defined by the **forward ODE** with hidden state $\boldsymbol{h}_t$:

$$\dot{\boldsymbol{h}}_t = \boldsymbol{W}\phi(\boldsymbol{h}_t), \quad \forall t \in [0, T] \tag{1}$$

  where $\boldsymbol{W} \in \mathbb{R}^{n \times n}$ and $\boldsymbol{W}_{ij} \overset{\text{iid}}{\sim} \mathcal{N}(0, 1/n)$.
- Given training data $\{(\boldsymbol{x}_i, y_i)\}_{i=1}^N$, we minimize the squared loss:

$$L(\boldsymbol{\theta}) = \sum_{i=1}^{N} \frac{1}{2}\left[f_{\boldsymbol{\theta}}(\boldsymbol{x}_i) - y_i\right]^2 = \frac{1}{2}\|\boldsymbol{u} - \boldsymbol{y}\|^2, \tag{2}$$

  where $\boldsymbol{u}$ is the prediction vector with $\boldsymbol{u}_i = f_\theta(\boldsymbol{x}_i)$.
- Gradients are computed via the **backward ODE** of the adjoint state $\boldsymbol{\lambda}_t$:

$$\dot{\boldsymbol{\lambda}}_t = -\,\mathbf{diag}(\phi'(\boldsymbol{h}_t))\boldsymbol{W}^\top, \quad \forall t \in [0, T] \tag{3}$$

Setup

- We study a Neural ODE $f_{\boldsymbol{\theta}}(\boldsymbol{x})$ defined by the **forward ODE** with hidden state $\boldsymbol{h}_t$:

$$\dot{\boldsymbol{h}}_t = \boldsymbol{W}\phi(\boldsymbol{h}_t), \quad \forall t \in [0, T] \tag{1}$$

  where $\boldsymbol{W} \in \mathbb{R}^{n \times n}$ and $\boldsymbol{W}_{ij} \overset{\text{iid}}{\sim} \mathcal{N}(0, 1/n)$.
- Given training data $\{(\boldsymbol{x}_i, y_i)\}_{i=1}^{N}$, we minimize the squared loss:

$$L(\boldsymbol{\theta}) = \sum_{i=1}^{N} \frac{1}{2}\left[f_{\boldsymbol{\theta}}(\boldsymbol{x}_i) - y_i\right]^2 = \frac{1}{2}\|\boldsymbol{u} - \boldsymbol{y}\|^2, \tag{2}$$

  where $\boldsymbol{u}$ is the prediction vector with $\boldsymbol{u}_i = f_{\theta}(\boldsymbol{x}_i)$.
- Gradients are computed via the **backward ODE** of the adjoint state $\boldsymbol{\lambda}_t$:

$$\dot{\boldsymbol{\lambda}}_t = -\,\mathbf{diag}(\phi'(\boldsymbol{h}_t))\boldsymbol{W}^\top, \quad \forall t \in [0, T] \tag{3}$$

- Gradient with respect to weights:

$$\nabla_{\boldsymbol{W}} f_{\boldsymbol{\theta}}(\boldsymbol{x}) = \int_0^T \boldsymbol{\lambda}_t \phi(\boldsymbol{h}_t)^\top \, dt \tag{4}$$

Setup

- We study a Neural ODE $f_{\boldsymbol{\theta}}(\boldsymbol{x})$ defined by the **forward ODE** with hidden state $\boldsymbol{h}_t$:

$$\dot{\boldsymbol{h}}_t = \boldsymbol{W}\phi(\boldsymbol{h}_t), \quad \forall t \in [0, T] \tag{1}$$

  where $\boldsymbol{W} \in \mathbb{R}^{n \times n}$ and $\boldsymbol{W}_{ij} \overset{\text{iid}}{\sim} \mathcal{N}(0, 1/n)$.
- Given training data $\{(\boldsymbol{x}_i, y_i)\}_{i=1}^{N}$, we minimize the squared loss:

$$L(\boldsymbol{\theta}) = \sum_{i=1}^{N} \frac{1}{2}\left[f_{\boldsymbol{\theta}}(\boldsymbol{x}_i) - y_i\right]^2 = \frac{1}{2}\|\boldsymbol{u} - \boldsymbol{y}\|^2, \tag{2}$$

  where $\boldsymbol{u}$ is the prediction vector with $\boldsymbol{u}_i = f_{\theta}(\boldsymbol{x}_i)$.
- Gradients are computed via the **backward ODE** of the adjoint state $\boldsymbol{\lambda}_t$:

$$\dot{\boldsymbol{\lambda}}_t = -\,\mathbf{diag}(\phi'(\boldsymbol{h}_t))\boldsymbol{W}^{\top}, \quad \forall t \in [0, T] \tag{3}$$

- Gradient with respect to weights:

$$\nabla_{\boldsymbol{W}} f_{\boldsymbol{\theta}}(\boldsymbol{x}) = \int_0^T \boldsymbol{\lambda}_t \phi(\boldsymbol{h}_t)^{\top}\, dt \tag{4}$$

- Parameters are updated by gradient descent:

$$\boldsymbol{\theta}^{(k+1)} = \boldsymbol{\theta}^{(k)} - \eta \nabla_{\boldsymbol{\theta}} L(\boldsymbol{\theta}^{(k)}) \tag{5}$$

Well-Posedness and Gradient Computation

Optimize-then-discretize: Use numerical solvers to compute gradients by solving the continuous forward and backward ODEs.

Well-Posedness and Gradient Computation

Optimize-then-discretize: Use numerical solvers to compute gradients by solving the continuous forward and backward ODEs.

Discretize-then-optimize: Discretize the ODE using Euler's method, treating the system as a finite-depth ResNet f_θ^L with shared parameters:

$$\boldsymbol{h}^\ell = \boldsymbol{h}^{\ell-1} + \Delta t \cdot \boldsymbol{W}\phi(\boldsymbol{h}^{\ell-1}), \quad \text{where } \Delta t = T/L \tag{6}$$

Well-Posedness and Gradient Computation

Optimize-then-discretize: Use numerical solvers to compute gradients by solving the continuous forward and backward ODEs.

Discretize-then-optimize: Discretize the ODE using Euler's method, treating the system as a finite-depth ResNet f_θ^L with shared parameters:

$$\boldsymbol{h}^\ell = \boldsymbol{h}^{\ell-1} + \Delta t \cdot \boldsymbol{W}\phi(\boldsymbol{h}^{\ell-1}), \quad \text{where } \Delta t = T/L \tag{6}$$

Proposition 1

Given $T < \infty$, if ϕ is Lipschitz continuous, then the forward and backward ODEs have unique solutions $\boldsymbol{h}_t$ and $\boldsymbol{\lambda}_t$ for all $t \in [0, T]$ and $\boldsymbol{x} \in \mathbb{R}^d$, almost surely over random initialization. Moreover, $\boldsymbol{\lambda}_t(\boldsymbol{x}) = \partial f(\boldsymbol{x}; \boldsymbol{\theta})/\partial \boldsymbol{h}_t$ is the solution to the backward ODE.

Well-Posedness and Gradient Computation

Optimize-then-discretize: Use numerical solvers to compute gradients by solving the continuous forward and backward ODEs.

Discretize-then-optimize: Discretize the ODE using Euler's method, treating the system as a finite-depth ResNet f_θ^L with shared parameters:

$$\boldsymbol{h}^\ell = \boldsymbol{h}^{\ell-1} + \Delta t \cdot \boldsymbol{W} \phi(\boldsymbol{h}^{\ell-1}), \quad \text{where } \Delta t = T/L \tag{6}$$

Proposition 1

Given $T < \infty$, if ϕ is Lipschitz continuous, then the forward and backward ODEs have unique solutions $\boldsymbol{h}_t$ and $\boldsymbol{\lambda}_t$ for all $t \in [0, T]$ and $\boldsymbol{x} \in \mathbb{R}^d$, almost surely over random initialization. Moreover, $\boldsymbol{\lambda}_t(\boldsymbol{x}) = \partial f(\boldsymbol{x}; \boldsymbol{\theta}) / \partial \boldsymbol{h}_t$ is the solution to the backward ODE.

Proposition 2

If, in addition, ϕ' is Lipschitz continuous, then the following holds a.s. over random initialization:

$$\left\| \nabla_{\boldsymbol{\theta}} f^L(\boldsymbol{x}) - \nabla_{\boldsymbol{\theta}} f(\boldsymbol{x}) \right\| \leq C L^{-1} \tag{7}$$

where $C > 0$ is a constant depending on the Lipschitz constants, time horizon T, and $\|\boldsymbol{x}\|$.

NTK of Neural ODEs

- Previous studies have shown the training dynamics of predictions $\boldsymbol{u}^k$ can be approximated by:

$$\boldsymbol{u}^{k+1} - \boldsymbol{y} \approx (\boldsymbol{I} - \eta \boldsymbol{H}^k)(\boldsymbol{u}^k - \boldsymbol{y}), \tag{8}$$

where $\boldsymbol{H}^k \in \mathbb{R}^{N \times N}$ is the NTK Gram matrix:

$$K_{\boldsymbol{\theta}}(\boldsymbol{x}, \bar{\boldsymbol{x}}) := \langle \nabla_{\boldsymbol{\theta}} f(\boldsymbol{x}; \boldsymbol{\theta}), \nabla_{\boldsymbol{\theta}} f(\bar{\boldsymbol{x}}; \boldsymbol{\theta}) \rangle \tag{9}$$

NTK of Neural ODEs

- Previous studies have shown the training dynamics of predictions $\boldsymbol{u}^k$ can be approximated by:

$$\boldsymbol{u}^{k+1} - \boldsymbol{y} \approx (\boldsymbol{I} - \eta \boldsymbol{H}^k)(\boldsymbol{u}^k - \boldsymbol{y}), \tag{8}$$

  where $\boldsymbol{H}^k \in \mathbb{R}^{N \times N}$ is the NTK Gram matrix:

$$K_{\boldsymbol{\theta}}(\boldsymbol{x}, \bar{\boldsymbol{x}}) := \langle \nabla_{\boldsymbol{\theta}} f(\boldsymbol{x}; \boldsymbol{\theta}), \nabla_{\boldsymbol{\theta}} f(\bar{\boldsymbol{x}}; \boldsymbol{\theta}) \rangle \tag{9}$$

- Neural ODE $f_{\boldsymbol{\theta}}$ can be viewed as the limit of a finite-depth ResNet $f_{\boldsymbol{\theta}}^L$ as $L \to \infty$.
- However, the NTK may depend on the order of limits:

$$\lim_{L\to\infty} \lim_{n\to\infty} f_{\boldsymbol{\theta}}^L \neq \lim_{n\to\infty} \lim_{L\to\infty} f_{\boldsymbol{\theta}}^L \tag{10}$$

NTK of Neural ODEs

- Previous studies have shown the training dynamics of predictions $\boldsymbol{u}^k$ can be approximated by:

$$\boldsymbol{u}^{k+1} - \boldsymbol{y} \approx (\boldsymbol{I} - \eta \boldsymbol{H}^k)(\boldsymbol{u}^k - \boldsymbol{y}), \tag{8}$$

where $\boldsymbol{H}^k \in \mathbb{R}^{N \times N}$ is the NTK Gram matrix:

$$K_{\boldsymbol{\theta}}(\boldsymbol{x}, \bar{\boldsymbol{x}}) := \langle \nabla_{\boldsymbol{\theta}} f(\boldsymbol{x}; \boldsymbol{\theta}), \nabla_{\boldsymbol{\theta}} f(\bar{\boldsymbol{x}}; \boldsymbol{\theta}) \rangle \tag{9}$$

- Neural ODE $f_{\boldsymbol{\theta}}$ can be viewed as the limit of a finite-depth ResNet $f_{\boldsymbol{\theta}}^L$ as $L \to \infty$.
- However, the NTK may depend on the order of limits:

$$\lim_{L \to \infty} \lim_{n \to \infty} f_{\boldsymbol{\theta}}^L \neq \lim_{n \to \infty} \lim_{L \to \infty} f_{\boldsymbol{\theta}}^L \tag{10}$$

- Illustrative example:

$$a_{n,\ell} := \frac{n}{\ell + n}, \quad \lim_{\ell \to \infty} \lim_{n \to \infty} a_{n,\ell} = 1, \quad \lim_{n \to \infty} \lim_{\ell \to \infty} a_{n,\ell} = 0 \tag{11}$$

NTK of Neural ODEs

- Previous studies have shown the training dynamics of predictions $\boldsymbol{u}^k$ can be approximated by:

$$\boldsymbol{u}^{k+1} - \boldsymbol{y} \approx (\boldsymbol{I} - \eta \boldsymbol{H}^k)(\boldsymbol{u}^k - \boldsymbol{y}), \tag{8}$$

where $\boldsymbol{H}^k \in \mathbb{R}^{N\times N}$ is the NTK Gram matrix:

$$K_{\boldsymbol{\theta}}(\boldsymbol{x}, \bar{\boldsymbol{x}}) := \langle \nabla_{\boldsymbol{\theta}} f(\boldsymbol{x}; \boldsymbol{\theta}), \nabla_{\boldsymbol{\theta}} f(\bar{\boldsymbol{x}}; \boldsymbol{\theta}) \rangle \tag{9}$$

- Neural ODE $f_{\boldsymbol{\theta}}$ can be viewed as the limit of a finite-depth ResNet $f_{\boldsymbol{\theta}}^L$ as $L \to \infty$.
- However, the NTK may depend on the order of limits:

$$\lim_{L\to\infty} \lim_{n\to\infty} f_{\boldsymbol{\theta}}^L \neq \lim_{n\to\infty} \lim_{L\to\infty} f_{\boldsymbol{\theta}}^L \tag{10}$$

- Illustrative example:

$$a_{n,\ell} := \frac{n}{\ell + n}, \quad \lim_{\ell\to\infty} \lim_{n\to\infty} a_{n,\ell} = 1, \quad \lim_{n\to\infty} \lim_{\ell\to\infty} a_{n,\ell} = 0 \tag{11}$$

Theorem 1 (NTK Convergence)

Suppose ϕ and ϕ' are Lipschitz. Then as width $n \to \infty$, the NTK $K_{\boldsymbol{\theta}}$ of the Neural ODE $f_{\boldsymbol{\theta}}$ converges almost surely to a deterministic kernel:

$$K_{\boldsymbol{\theta}} \to K_\infty, \quad \text{as } n \to \infty, \tag{12}$$

where K_∞ equals the limit of NTKs K_∞^L for ResNets as depth $L \to \infty$.

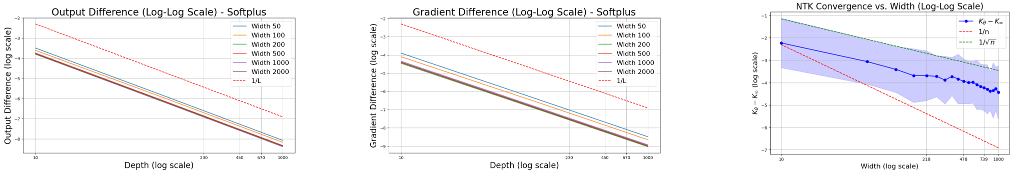


Figure: Comparison of Neural ODEs and ResNets: With Softplus (smooth), output, gradient, and NTK differences decay as $1/L$.

Empirical Validation of Well-Posedness and NTK Convergence

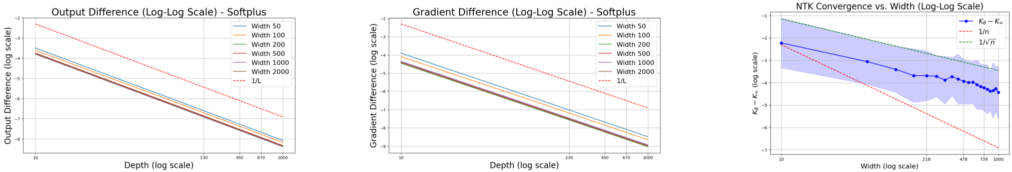


Figure: Comparison of Neural ODEs and ResNets: With Softplus (smooth), output, gradient, and NTK differences decay as $1/L$.

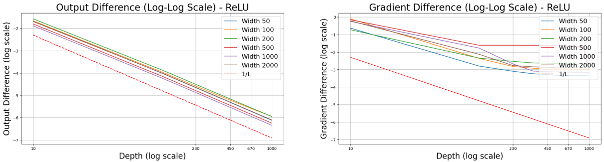


Figure: Comparison of Neural ODEs and ResNets with ReLU (non-smooth) activations: Output difference shows $1/L$ convergence, while gradient difference remains constant as depth L increases.

- Previous works have shown a strictly positive definite (SPD) limiting NTK K_∞ is sufficient to guarantee global convergence under gradient descent.

Spectral Properties of the NTK

- Previous works have shown a strictly positive definite (SPD) limiting NTK K_∞ is sufficient to guarantee global convergence under gradient descent.
- However, the SPD property of the finite-depth NTK K_∞^L does not ensure that K_∞ remains SPD as $L \to \infty$. The spectrum may degenerate with increasing depth.

- Previous works have shown a strictly positive definite (SPD) limiting NTK K_∞ is sufficient to guarantee global convergence under gradient descent.
- However, the SPD property of the finite-depth NTK K_∞^L does not ensure that K_∞ remains SPD as $L \to \infty$. The spectrum may degenerate with increasing depth.
- The NTK of the Neural ODE admits an **integral form**:

$$K_\infty = \Sigma^{T,T} + \int_0^T \int_0^T \Sigma^{t,s} K^{t,s} \, dt \, ds + \Sigma^{0,0} K^{0,0}$$

where $\Sigma^{t,s}$ and $K^{t,s}$ encode covariance and kernel terms at time t, s.

- Previous works have shown a strictly positive definite (SPD) limiting NTK K_∞ is sufficient to guarantee global convergence under gradient descent.
- However, the SPD property of the finite-depth NTK K_∞^L does not ensure that K_∞ remains SPD as $L \to \infty$. The spectrum may degenerate with increasing depth.
- The NTK of the Neural ODE admits an **integral form**:

$$K_\infty = \Sigma^{T,T} + \int_0^T \int_0^T \Sigma^{t,s} K^{t,s} \, dt \, ds + \Sigma^{0,0} K^{0,0}$$

where $\Sigma^{t,s}$ and $K^{t,s}$ encode covariance and kernel terms at time t, s.

Proposition 3

Suppose ϕ and ϕ' are Lipschitz continuous. If ϕ is nonlinear and non-polynomial, then the limiting NTK K_∞ is strictly positive definite.

Global Convergence for Neural ODEs

Assumption 1

Let $\{\boldsymbol{x}_i, y_i\}_{i=1}^N$ *be a training set. Assume*

1. *Training set:* $\boldsymbol{x}_i \in \mathbb{S}^{d-1}$ *and* $\boldsymbol{x}_i \neq \boldsymbol{x}_j$ *for all* $i \neq j$; $|y_i| = \mathcal{O}(1)$,
2. *Smoothness:* ϕ *and* ϕ' *are Lipschitz continuous, respectively,*
3. *Nonlinearity:* ϕ *is nonlinear and non-polynomial.*

Global Convergence for Neural ODEs

Assumption 1

Let $\{\boldsymbol{x}_i, y_i\}_{i=1}^N$ *be a training set. Assume*

1. *Training set:* $\boldsymbol{x}_i \in \mathbb{S}^{d-1}$ *and* $\boldsymbol{x}_i \neq \boldsymbol{x}_j$ *for all* $i \neq j$; $|y_i| = \mathcal{O}(1)$,
2. *Smoothness:* ϕ *and* ϕ' *are Lipschitz continuous, respectively,*
3. *Nonlinearity:* ϕ *is nonlinear and non-polynomial.*

Theorem 2

Suppose Assumption 1 holds and the learning rate η *is chosen s.t.* $0 < \eta \leq 1/\|\boldsymbol{X}\|^2$. *Then for any* $\delta > 0$, *there exists a natural number* n_δ *s.t. for all widths* $n \geq n_\delta$ *the following results hold with probability at least* $1 - \delta$ *over random initialization:*

1. *The parameters* $\boldsymbol{\theta}^k$ *stay in a neighborhood of* $\boldsymbol{\theta}^0$, *i.e.,*

$$\|\boldsymbol{\theta}^k - \boldsymbol{\theta}^0\| \leq C\|\boldsymbol{X}\|\sqrt{L(\boldsymbol{\theta}_0)}/\lambda_0. \tag{13}$$

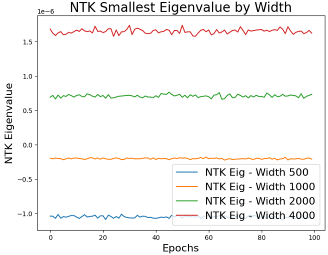
2. *The loss function* $L(\boldsymbol{\theta}_k)$ *consistently decreases to zero at an exponential rate, i.e.,*

$$L(\boldsymbol{\theta}_k) \leq (1 - \eta\boldsymbol{\lambda}_0)^k L(\boldsymbol{\theta}_0), \tag{14}$$

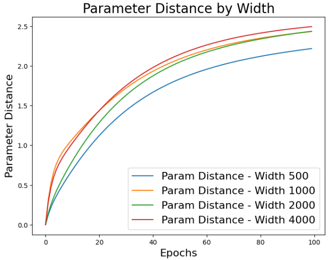
where $\lambda_0 := \lambda_{\min}(K_\infty) > 0$, *and the constant* $C > 0$ *only depends on Lipschitz coefficients and* T.

Empirical Validation of Global Convergence

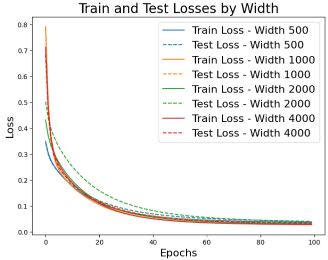
- We examine NTK spectra, parameter stability, and loss convergence across different model widths.



(a) NTK eigenvalues

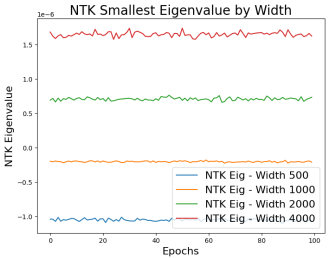


(b) Distance from initialization

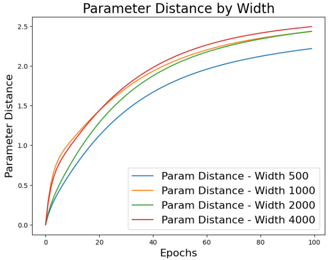


(c) Train vs. test loss

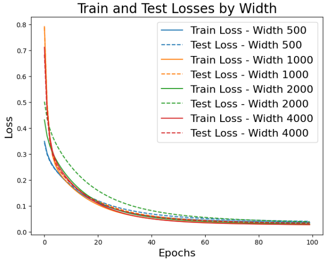
- We examine NTK spectra, parameter stability, and loss convergence across different model widths.



(a) NTK eigenvalues



(b) Distance from initialization



(c) Train vs. test loss

- Additional experiments: time horizon effects, quadratic activations, solver comparisons (adaptive vs. fixed step). See the **poster** or **paper** for details.

Future Work & Collaboration

Open Directions

- **Convergence Rates:** Derive explicit rates as width/depth grow in Neural ODEs.
- **Solver Design:** Design efficient ODE solvers guided by gradient alignment conditions.
- **Beyond GD:** Explore SGD, momentum, and adaptive optimization methods.
- **Feature Learning:** Analyze training and generalization beyond the lazy training regime.
- **Broader Models:** Extend theory to Transformers, ResNets, and state-space models.

Contact Information:

- Email: tgao9@depaul.edu
- Web: https://gaotx-cs.github.io/