Stability Regions Control Gradient Flow: Why Neural ODEs Need Extended-Stability Solvers

Gavin Lee Goodship[1], Stephen O'Sullivan[1], Luis Miralles-Pechuán[1,2]
[1]School of Computer Science, Technological University Dublin, Dublin, Ireland
[2]EUT+ Data Science Institute, European University of Technology, European Union
Email: {D23126880, Luis.Miralles, Stephen.OSullivan}@TUDublin.ie

***Abstract*—Neural ordinary differential equations (ODEs) provide a principled continuous-depth formulation, but their practical use is often limited by slow and numerically fragile training due to the weaknesses of standard ODE solvers. We introduce extended-stability Runge–Kutta methods, explicit fixed-step solvers that admit much larger step sizes without numerical blow-up in negative-real-axis dominated regimes than classical schemes such as RK4 or adaptive methods such as Dormand–Prince. By enlarging the real-axis stability region, these solvers traverse a fixed integration horizon with far fewer steps, yielding deterministic computation, reduced function-evaluation cost, and smoother optimization dynamics. Across CIFAR-10, CIFAR-100, and Tiny-ImageNet, extended-stability solvers match final accuracy (typically within 1–2%) while improving gradient-flow conditioning, reducing gradient clipping rates (e.g., from 60–100% to 0–25% on CIFAR-10, depending on the horizon). The 15-stage variant achieves a $14\times$ wall-clock speedup over Dormand–Prince. Overall, our results show that stability-region geometry, rather than solver order or adaptivity, governs gradient-flow conditioning under fixed integration horizons, making extended-stability methods an effective tool for accelerating training and improving optimization conditioning in Neural ODEs.**

***Index Terms*—Neural Ordinary Differential Equations, Neural ODE, Continuous-Depth Networks, Numerical Solvers, ESRK Methods, Stability Analysis.**

I. Introduction

Neural ordinary differential equations (Neural ODEs) [1] represent the depth of a network in continuous time, with forward and backward passes implemented through numerical ODE solvers. Consequently, Neural ODE behavior is determined not only by the learned vector field but also by the numerical integrator used during training. The solver defines the discrete dynamics, influences stability, and governs computational cost, yet is often treated as a secondary implementation detail. This leads to well-known limitations. Adaptive solvers such as Dormand–Prince 5(4) (DoPri5), commonly used in Neural ODE libraries, exhibit highly variable numbers of function evaluations (NFEs), resulting in unpredictable compute and unstable optimization dynamics. Fixed-step solvers such as Euler or RK4 provide deterministic execution, but possess small stability regions (e.g., RK4 is stable only for $z \in [-2.8, 0]$ on the real axis), forcing small step sizes when the learned dynamics become stiff. As a result, models trained under different solvers often exhibit different behaviors, indicating that the choice of solver affects learning itself rather than only the evaluation.

To support reproducibility, the full training and evaluation code for all experiments is publicly available at https://doi.org/10.5281/zenodo.19189552, including implementation details and coefficients for all extended-stability Runge–Kutta (ESRK) schemes used.

a) Key insight: We show that this behavior is governed primarily by a numerical property of the solver: the geometry of its stability region. The stability region determines how perturbations and gradients propagate over time, directly influencing gradient conditioning during backpropagation. Although solver order and adjoint methods have been widely studied, the role of the stability polynomial $R(z)$ and its real-axis stability interval has received little attention in the literature on Neural ODE.

b) Approach: To isolate the effect of stability-region geometry, we employ extended-stability Runge–Kutta (ESRK) methods [2], [3]. ESRKs remain explicit and fixed-cost, but their real-axis stability intervals grow approximately quadratically with stage count. This allows us to vary the stability-region size without changing model architecture, optimization settings, or solver order, making ESRKs a controlled instrument for probing the relationship between stability and trainability.

c) Findings and implications: Across CIFAR-10, CIFAR-100, and Tiny-ImageNet, and across integration horizons ($\Delta T = 10$–30), we find that increasing stability-region size preserves final accuracy while consistently reducing spectral amplification and improving gradient-flow conditioning. These results demonstrate that stability-region geometry, rather than solver order, governs trainability in Neural ODEs. Solver choice should therefore be treated as a first-class numerical design parameter in continuous-depth learning systems.

To our knowledge, this is the first work to empirically isolate solver stability-region geometry as a controlling factor for gradient-flow conditioning in Neural ODE training, independently of solver order, adaptivity, or model capacity.

II. Related Work

We review prior work on Neural ODEs, solver dependence in continuous-depth learning, and stabilized explicit integrators, and identify a gap in understanding the role of stability-region geometry in optimization.

a) Neural ODEs: Neural ordinary differential equations (Neural ODEs) [1] model the depth of the network in continuous time and are trained via numerical integration and adjoint sensitivities. They have been applied to generative modeling [4], irregular time-series [5], stochastic and physics-informed systems [6].

b) Solver dependence: Previous work has shown that Neural ODE training is highly sensitive to the choice of numerical solver. Fixed-step methods (Euler, RK2, RK4) offer predictable computation but possess small stability regions, forcing small step sizes when dynamics become stiff. Adaptive solvers such as Dormand–Prince and Tsit5 [7], [8] provide local error control but introduce variable numbers of function evaluations (NFEs), irregular memory access, and unstable optimization dynamics. This trade-off between stability and predictability is a central limitation in continuous-depth learning [9], [10].

c) Stabilized explicit integrators: In numerical analysis, Chebyshev-based stabilized explicit schemes such as RKC and ROCK were developed to handle stiff negative-real-axis dynamics while remaining explicit [11], [12], [13]. Their real-axis stability intervals scale approximately as $L_- \propto s^2$ with stage count s, enabling large stable steps without implicit solves. Low-storage formulations further improve efficiency on modern hardware [14], [15]. Extended–stability Runge-Kutta (ESRK) methods belong to this class.

d) Gap and contribution: Despite their alignment with Neural ODE requirements (explicitness, fixed cost, and stability at large step sizes), the geometry of the stability region has not been treated as a modeling or optimization parameter in Neural ODE research. Existing work emphasizes solver order, adjoint memory reuse, or adaptive step-size control, but does not examine how the stability polynomial $R(z)$ governs gradient propagation. In this work, we use ESRK methods as a controlled instrument for varying stability-region size without changing model capacity, allowing us to isolate its effect on gradient-flow conditioning and learning dynamics.

A. Construction of Extended-Stability Runge–Kutta Schemes

Extended-stability explicit Runge–Kutta methods have been developed in numerical analysis as an efficient approach for integrating moderately stiff differential equations. Classical Runge–Kutta–Chebyshev (RKC) methods construct stability polynomials whose real-axis stability interval grows quadratically with the number of stages. In [2], a systematic development of high-order Chebyshev stability polynomials was presented, providing a principled framework for designing explicit methods with expanded stability domains. For explicit Runge–Kutta methods, numerical stability is characterized through the stability function $R(z)$ applied to the linear test equation $y' = \lambda y$, where $z = h\lambda$. The stability region is defined as $\{z : |R(z)| < 1\}$, and its extent along the negative real axis is particularly important for Neural ODEs, where learned dynamics often exhibit dissipative behavior. Classical methods such as RK4 have limited real-axis stability

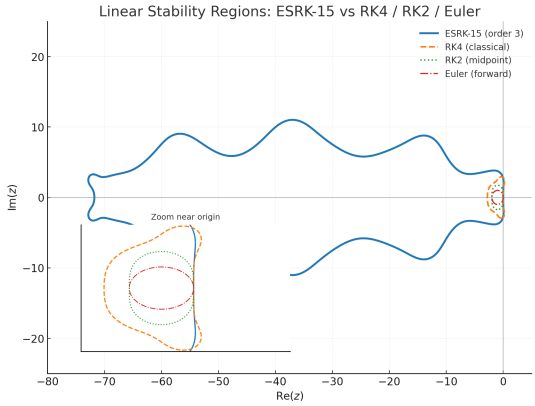

Fig. 1: Negative real-axis stability regions for Forward Euler, RK2, RK4, and ESRK-15. ESRK-15 achieves a much larger stability interval ($L_- \approx 72$ vs. 2.8 for RK4).

($L_- \approx 2.8$), constraining admissible step sizes and leading to increased computational cost in stiff regimes.

Extended-stability Runge–Kutta schemes enlarge this region while remaining explicit and fixed-cost. In particular, the length of the negative real-axis stability interval scales as

$$L_- \propto s^2,$$

where s denotes the number of stages.

In this work, we employ schemes constructed using the Runge–Kutta–Gegenbauer (RKG) framework introduced by [3]. This formulation generalizes Chebyshev-based stability polynomials by introducing a Gegenbauer parameter ν, which controls the shape of the stability region in both the real and imaginary directions. The stability polynomial of degree L is given by

$$R_L(z) = \frac{G_L^{(\nu)}(1 + \alpha z)}{G_L^{(\nu)}(1)},$$

where $G_L^{(\nu)}$ denotes the Gegenbauer polynomial, α is a scaling factor determining the extent of the real-axis stability interval, and ν governs the width of the stability region. Once the stability polynomial is defined, the corresponding Runge–Kutta scheme is obtained via a sequence of forward Euler stages determined by the polynomial roots, with coefficients chosen to satisfy the required order conditions. Crucially, varying s or the Gegenbauer parameter ν does not change the solver order, model architecture, or optimization procedure; instead, it provides a controlled mechanism for modifying the stability polynomial and hence the stability-region geometry. This enables us to isolate the effect of stability geometry on training dynamics. We therefore employ ESRK-15 and ESRK-21 as a controlled family of solvers sharing the same polynomial construction but differing in stability envelope and implementation. This allows us to investigate how stability-polynomial structure and stage realization induce distinct geometric and optimization regimes in Neural ODE training, even under matched predictive performance.

III. EXPERIMENTS

Our experiments test a single hypothesis: the stability region of the ODE solver controls the conditioning of gradient flow in Neural ODEs. We evaluate the solvers in CIFAR-10, CIFAR-100, and Tiny-ImageNet, and across multiple integration horizons (ΔT=10, 20, 30), while maintaining the architecture, optimizer, and training schedule fixed. All models contain fewer than 100k parameters, and all reported values are averaged over three seeds.

a) Solvers and Fairness Protocol: We compare RK4 and Dormand–Prince 5(4) (DoPri5) as explicit and adaptive baseline solvers and use ESRK-15 and ESRK-21 to vary stability-region size without changing solver order or model capacity.

b) Comparison regimes: All comparisons are performed under the following fairness conditions:

- A fixed integration horizon ΔT, ensuring identical continuous-time depth across solvers;
- Deterministic NFE for fixed-step solvers (RK4, ESRK-15, ESRK-21), which is matched by construction;
- Reported NFE and wall-clock time for adaptive DoPri5, reflecting the intrinsic computational overhead of bounded adaptivity.

DoPri5 is evaluated in a bounded-adaptive configuration to prevent runaway step subdivision; its resulting NFE and runtime are reported rather than enforced to match fixed-step methods.

c) Integration Horizon Matching: All solvers are compared under a fixed integration horizon of $\Delta T = 30$. To achieve this while preserving each solver's stability regime, we use:

- **ESRK-15**: one solver step with step size $h = 30$ ($1\times30 = \Delta T$),
- **RK4**: four fixed steps with step size $h = 7.5$ ($4 \times 7.5 = \Delta T$),
- **DoPri5 (adaptive-bounded)**: two macro-steps with a nominal step size $h = 15$ ($2\times15 = \Delta T$), using embedded error control and restricted adaptivity.

This configuration ensures that all solvers traverse the same effective continuous-time interval, while differing only in their stability mechanisms and step discretization strategies. We do not tune solvers for advantage; all settings follow classical stability limits and are consistent with prior Neural ODE literature.

d) On fairness and DoPri5 adaptivity: To ensure a fair comparison, we employ a bounded-adaptive variant of Dormand–Prince 5(4). The solver retains its local error controller but is restricted to step sizes in the range $[0.05, 1.0]$ within each macro-step (h=15, steps= 2, ΔT=30). This configuration allows moderate sub-stepping in stiff regions typically yielding 10–15k function evaluations per epoch while preventing runaway subdivision beyond 40k NFEs. The reported 10–18$\times$ wall-clock overhead therefore represents a realistic but bounded adaptive baseline. Under fully unconstrained tolerance settings, the cost can increase by another 2–4$\times$, depending on the stiffness of the dataset and the clipping frequency.

TABLE I: Common training and model hyperparameters used across all experiments.

Component	**Setting**
Dataset	CIFAR-10 / CIFAR-100 / Tiny-ImageNet
Image resolution	32×32 (CIFAR), 64×64 (Tiny-ImageNet)
Model backbone	`TinyNode(width=64)`
ODEs solver	RK4, ESRK-15, ESRK-21, DoPri5
Activation function	SiLU
Normalization	None or GroupNorm(8) (when specified)
Classifier head	Linear(*width*, n_{classes}), where $n_{\text{classes}} \in \{10, 100, 200\}$
Optimizer	AdamW
Initial learning rate	3×10^{-4}
Weight decay	5×10^{-4}
Batch size	128
Schedule	Cosine annealing with 5-epoch warm-up
Label smoothing	0.0 (disabled)
Gradient clipping	10 (unless stated)
Training epochs	150 (CIFAR), 80 (Tiny-ImageNet)
Spectral diagnostics	JVP-based power iteration (5 steps)

e) Model and Training Configuration: All models are trained under a unified optimization and regularization protocol to ensure that differences in performance arise from the behavior of the ODE solver rather than dataset-specific tuning or auxiliary heuristics. We apply no dataset-specific hyperparameter searches and avoid additional stabilization techniques such as label smoothing, mixup, CutMix, stochastic depth, or exponential-moving averages. Apart from standard random crop and horizontal flip, no further data augmentation is used. The learning rate, weight decay, batch size, and scheduling remain identical in all solvers and datasets.

f) Hardware and software configuration: All experiments were conducted on a single NVIDIA GeForce RTX 4050 GPU (6 GB memory) using PyTorch 2.7.1 with CUDA support. The system used NVIDIA driver version 535.247.01 with CUDA runtime 12.2, and cuDNN 9.1 was enabled. Automatic mixed-precision (AMP) training was used consistently across all solvers. Experiments were run under a Linux operating system.

g) Training protocol: All models use direct differentiation through the solver. We do not use the continuous-time adjoint method, avoiding adjoint–forward mismatch and ensuring gradients correspond exactly to the discretized dynamics used during training.

h) Gradient-Flow Metrics: To analyze how stability-region size affects optimization, we track four diagnostics: the pre-clip gradient magnitude, the gradient clipping rate, the cosine alignment between successive gradient steps, and a Jacobian spectral proxy $\hat{\sigma}$. These metrics separate predictive

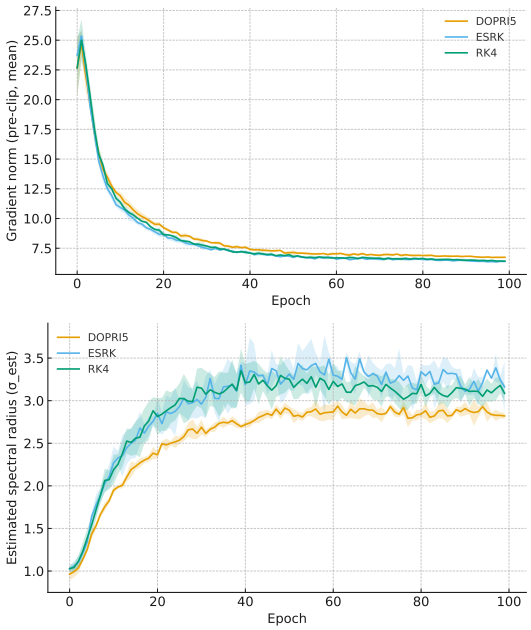

Fig. 2: Gradient-flow diagnostics on CIFAR-10. Top: pre-clip gradient norm $\|g_{\text{pre}}\|$; Bottom: spectral proxy $\hat{\sigma}$.

TABLE II: Final validation accuracy at ΔT=30 across datasets (3 seeds).

Solver	CIFAR-10	CIFAR-100	Tiny-ImageNet
RK4	83.2 ± 0.5	49.3 ± 0.7	37.6 ± 0.3
DoPri5	83.0 ± 0.7	49.0 ± 0.6	37.8 ± 0.2
ESRK-15 (ours)	$\mathbf{82.1 \pm 0.9}$	$\mathbf{49.8 \pm 0.6}$	$\mathbf{38.0 \pm 0.3}$

accuracy from optimization conditioning, allowing us to characterize how solver stability influences gradient propagation during training.

Metric definitions We measure the pre-clip gradient norm $\|g_t\|_2$ (after AMP unscale). The clip rate is the fraction of steps with $\|g_t\|_2 > \tau$. Cosine alignment is $\langle g_t, g_{t-1}\rangle/(\|g_t\|_2\|g_{t-1}\|_2)$ averaged over steps. The spectral proxy $\hat{\sigma}$ is estimated per batch by 5-step power iteration using JVPs and averaged per epoch.

i) Baseline: CIFAR-10 at $\Delta T = 30$: We first evaluate RK4, DoPri5, and ESRK-15 in CIFAR-10 at a matched integration horizon ΔT=30 using a 64-channel `TinyNODE` model (76k parameters). Model configurations are provided in Section III-0e. ESRK-15 achieves comparable final accuracy to RK4 and DoPri5 while requiring fixed, predictable compute. ESRK-15 exhibits smoother gradient dynamics (lower clip rate, bounded $\hat{\sigma}$), whereas RK4 and DoPri5 show oscillatory gradients and instability pressure; this can be seen in Figure 2.

j) Cross-Dataset Stability Scaling: CIFAR-100 and Tiny-ImageNet: Repeating the protocol on CIFAR-100 and Tiny-ImageNet yields the same outcome: ESRK-15 matches final accuracy while providing smoother gradient transport and bounded spectral growth at ΔT=30 (Figures 3, 4, and 5).

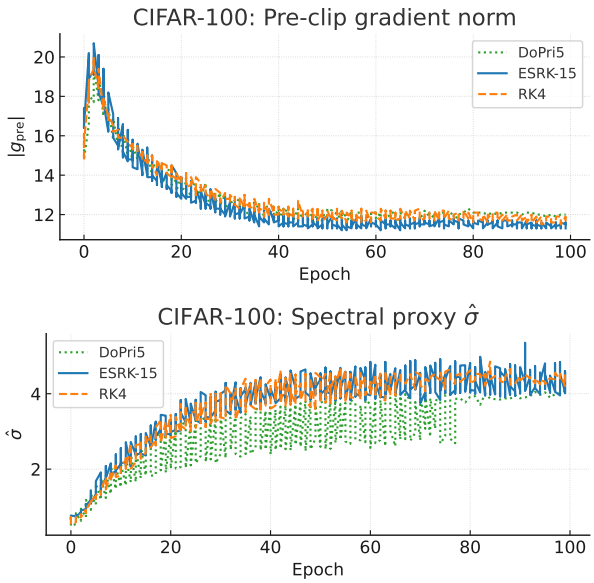

Fig. 3: Gradient-flow diagnostics on CIFAR-100. **Top:** pre-clip gradient norm $\|g_{\text{pre}}\|$ (mean±SD). **Bottom:** spectral proxy $\hat{\sigma}$. ESRK-15 yields smoother gradient dynamics and reduced spectral amplification compared to RK4 and DoPri5.

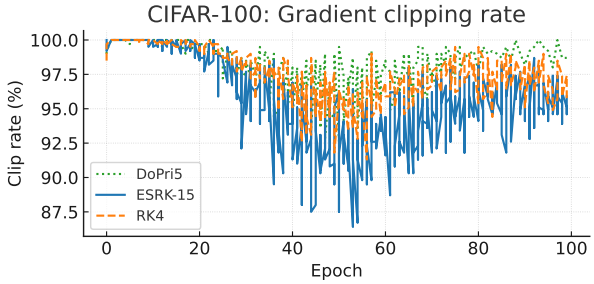

Fig. 4: Clip rate (%) vs. epoch on CIFAR-100. Clipping remains high under the fixed threshold ($\tau = 10$), indicating a clipping-dominated regime; ESRK-15 primarily improves spectral amplification and gradient coherence rather than eliminating clipping at this integration horizon.

k) Effect of Stability-Region Size: ESRK-15 vs ESRK-21: To isolate the effect of stability-region geometry, we compare ESRK-15 and ESRK-21, which share identical order and low-storage structure but differ in real-axis stability radius ($L_- \approx 72$ vs. $L_- \approx 144$). Importantly, both architectures achieve comparable accuracy across CIFAR-10, CIFAR-100, and Tiny-ImageNet, indicating that increasing the stage count does not change model expressivity. The differences we observe therefore reflect how gradients are transported through the ODE block, rather than which functions the model is capable of representing.

We examine two integration horizons. At $\Delta T = 10$ (a *moderate-gain* regime), both ESRK-15 and ESRK-21 operate well inside their respective stability regions. In this regime, gradient-flow statistics are nearly identical or clipping-

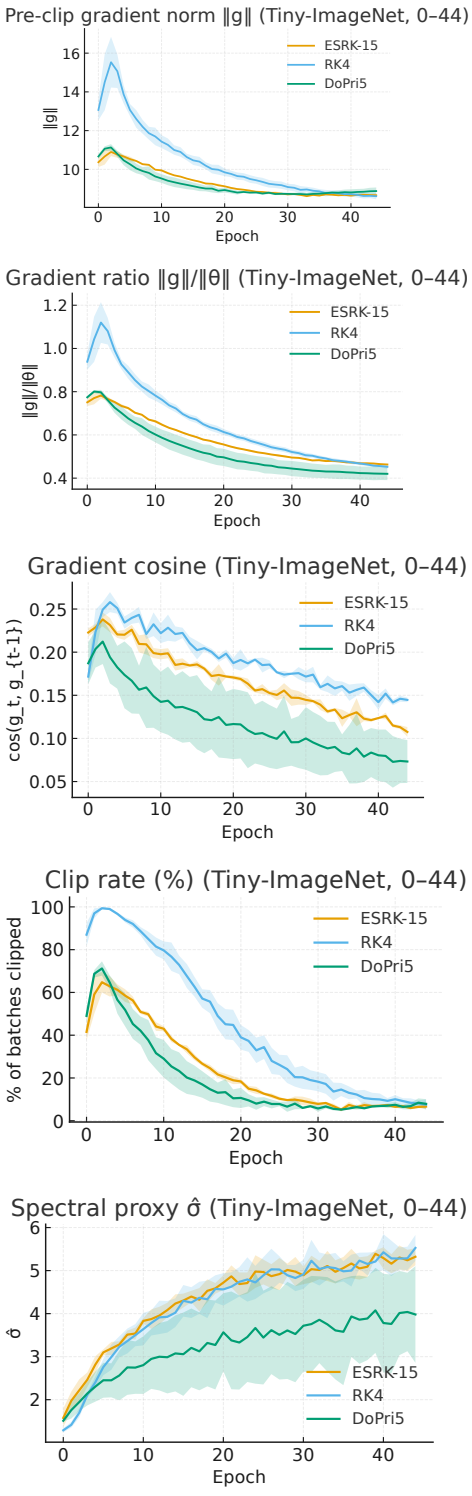

Fig. 5: Gradient-flow diagnostics on Tiny-ImageNet at $\Delta T = 30$. From top to bottom: pre-clip gradient norm, gradient-to-parameter ratio, cosine alignment, clip rate, and spectral proxy $\hat{\sigma}$. ESRK-15 exhibits smoother gradients, reduced clipping, and lower spectral amplification than RK4 and DoPri5.

Solver	$\|g_{\text{pre}}\|$	Clip Rate (%)	Cosine Align	$\hat{\sigma}$
ESRK-15	5.95	0.0	0.0059	0.96
ESRK-21	9.01	16.8	0.0026	0.43

TABLE III: **CIFAR-10,** $\Delta T = 10$ **(moderate regime).**

Solver	$\|g_{\text{pre}}\|$	Clip Rate (%)	Cosine Align	$\hat{\sigma}$
ESRK-15	7.19	0.26	0.0033	0.57
ESRK-21	8.79	13.25	0.0041	0.44

TABLE IV: **CIFAR-10,** $\Delta T = 20$ **(high-gain regime).**

dominated across datasets, confirming that increasing stage count does not introduce any measurable optimization advantage when the dynamics remain numerically well-conditioned (Tables III, V, and VII).

When the integration horizon increases to $\Delta T = 20$, training enters a *high-gain* regime in which the stability-region size becomes the dominant factor that governs gradient transport. In this setting, ESRK-21 maintains stronger gradients, reduces spectral amplification, and maintains a smoother cosine alignment relative to ESRK-15 across all datasets (Tables IV, VI and VIII). This demonstrates that increasing the extended-stability stage count improves gradient-flow conditioning precisely when the learned dynamics approach the stability boundary.

Across all datasets, increasing stage count does *not* change accuracy and yields little effect in stable regimes, but in high-gain regimes ($\Delta T = 20$) a larger stability-region radius consistently improves gradient-flow conditioning. This establishes the first empirical link between extended-stability Runge–Kutta design and gradient transport in Neural ODE training.

l) Gradient clipping: Unless otherwise stated, all experiments use gradient clipping with threshold $\tau = 10$. We verified that the solver-dependent ordering and stability-region effects reported in this work are robust to the choice of clipping threshold by evaluating both more aggressive clipping ($\tau = 5$) and weak clipping ($\tau = 20$). The relative conditioning trends remain unchanged across clipping levels, indicating that our conclusions are not driven by gradient clipping.

m) Synthesis of Experimental Findings: The empirical results across CIFAR-10, CIFAR-100, and Tiny-ImageNet provide consistent evidence that the size of the solver's stability region plays a central role in conditioning gradient flow during Neural ODEs training. ESRK-15 achieves an accuracy comparable to RK4 and bounded-adaptive DoPri5 while exhibiting improved optimization conditioning, reflected in lower clipping rates, reduced oscillatory gradients, and bounded spectral growth.

These effects arise despite all solvers integrating over the same continuous-time horizon and operating under identical architectural and training configurations, indicating that differences in gradient dynamics derive from numerical properties rather than model capacity or optimization strategy. The comparison between ESRK-15 and ESRK-21 further isolates the effect of stability-region geometry. When the integra-

Solver	$\|g_{\text{pre}}\|$	Clip Rate (%)	Cosine Align	$\hat{\sigma}$
ESRK-15	12.24	99.5	0.00125	1.02
ESRK-21	12.86	99.9	0.00246	0.71

TABLE V: **CIFAR-100,** $\Delta T = 10$ **(clipping-dominated regime).**

Solver	$\|g_{\text{pre}}\|$	Clip Rate (%)	Cosine Align	$\hat{\sigma}$
ESRK-15	15.07	100.0	0.00221	0.65
ESRK-21	15.83	100.0	0.00371	0.39

TABLE VI: **CIFAR-100,** $\Delta T = 20$ **(high-gain regime).**

Solver	$\|g_{\text{pre}}\|$	Clip Rate (%)	Cosine Align	$\hat{\sigma}$
ESRK-15	9.48	26.1	0.0045	8.64
ESRK-21	9.42	24.1	0.0046	9.34

TABLE VII: **Tiny-ImageNet,** $\Delta T = 10$ **(moderate regime).**

Solver	$\|g_{\text{pre}}\|$	Clip Rate (%)	Cosine Align	$\hat{\sigma}$
ESRK-15	9.51	25.3	0.0048	6.67
ESRK-21	9.38	20.8	0.0094	6.44

TABLE VIII: **Tiny-ImageNet,** $\Delta T = 20$ **(high-gain regime).**

tion horizon remains well within the stability region (e.g. $\Delta T = 10$), both methods display similar gradient statistics, suggesting that an increased stage count does not confer additional optimization benefits in well-conditioned regimes. However, when the integration horizon approaches the stability boundary ($\Delta T = 20$), ESRK-21 consistently yields more coherent gradients, lower spectral amplification, and reduced clipping.

These findings demonstrate that enlarging the stability radius enhances optimization specifically in high-gain regimes, reinforcing the conclusion that stability-region geometry is a dominant control parameter for optimization under fixed integration horizons in continuous-depth training. Although evaluated on compact vision models, the consistent trends across explicit, adaptive, and extended-stability solvers suggest that stability geometry is a general mechanism in Neural ODE training.

n) Computational efficiency and practical tradeoffs.: We evaluate wall-clock training time, computational cost (FLOPs), and throughput on Tiny-ImageNet at $\Delta T = 30$ (Table IX). ESRK-15 trains in approximately 98–100 seconds per epoch with deterministic NFE, yielding a $14\times$ speedup over bounded-adaptive DoPri5 (1 400–1 500 seconds), while using slightly fewer FLOPs than RK4 (0.58 vs. 0.62 GFLOPs). Although RK4 achieves higher throughput, it does so at the cost of severe gradient instability (near-100% clipping versus $\sim$25% for ESRK-15).

The NFE comparison (Table X) highlights the source of this overhead: ESRK-15 requires 600 function evaluations per epoch, whereas DoPri5 incurs $\sim$11 000 NFE due to adaptive subdivision, corresponding to an $18.3\times$ relative cost. ESRK-21 increases computational cost to $1.4\times$ ESRK-15 but provides additional gradient smoothing in high-gain regimes ($\Delta T \geq 20$). Overall, ESRK methods offer a favorable speed–stability tradeoff, combining predictable computational cost with improved gradient conditioning, and constitute practical fixed-step alternatives to both RK4 and adaptive solvers for Neural ODE training.

IV. Discussion

Our experiments reveal a consistent cross-dataset pattern: the stability region of the ODE solver governs gradient-flow conditioning in Neural ODE training. Across CIFAR-10, CIFAR-100, and Tiny-ImageNet, and across integration horizons $\Delta T \in \{10, 20, 30\}$, increasing the extended-stability stage count (ESRK-15 $\rightarrow$ ESRK-21) preserves predictive accuracy while producing smoother, more coherent, and better-conditioned gradients. This indicates that stage count does not act as a capacity parameter, but rather as a numerical mechanism controlling gradient transport.

a) Positioning relative to RK4 and DoPri.: At large integration horizons ($\Delta T = 30$), RK4 becomes unstable unless the step size is reduced, while adaptive solvers such as DoPri5 remain stable only by introducing highly variable and often prohibitive numbers of function evaluations. Once this limitation is established, comparisons at large step sizes are no longer meaningful for RK4 or DoPri. We therefore focus on controlled comparisons within the ESRK family to isolate the effect of stability-region geometry independent of architecture, optimizer, or parameter count.

b) Stability-region geometry and gradient transport.: For a discretized flow map $x_{t+h} = \Phi_h(x_t)$, the propagation of the gradient is governed by the local sensitivity $\partial \Phi_h / \partial x \approx R(h\lambda)$, where λ denotes Jacobian eigenvalues of the learned dynamics. The magnitude $|R(h\lambda)|$ determines whether the gradients are amplified, damped, or transported coherently. Enlarging the real-axis stability interval via increased ESRK stage count suppresses internal amplification within the stability region, yielding smoother spectral dynamics and improved gradient propagation.

c) Regime-dependent behavior.: The benefit of increasing stage count depends on the training regime. At moderate step sizes ($\Delta T = 10$), ESRK-15 already provides sufficient stability and ESRK-21 yields little additional benefit. At larger horizons ($\Delta T \geq 20$), where dynamics become stiff and gradients approach the stability boundary, ESRK-21 consistently reduces spectral amplification and improves gradient-flow conditioning. This regime dependence explains the observed task-specific effects, including reversals in more challenging datasets such as CIFAR-100.

d) Order versus stability.: Although ESRK-15 is third-order, while RK4 and DoPri5 are higher-order, all solvers achieve comparable final accuracy. This indicates that local truncation error is not the limiting factor in Neural ODE training. Instead, gradient conditioning is governed by the geometry of the stability region. Within the ESRK family, increasing stage count improves gradient-flow metrics without affecting accuracy, directly confirming that stability, not order,

TABLE IX: **Runtime and throughput on Tiny-ImageNet (ΔT=30).** ESRK-15 is 14× faster than adaptive DoPri5 while exhibiting improved gradient stability.

Solver	Time per Epoch	FLOPs/Forward	Throughput (img/s)
ESRK-15 (ours)	∼98–100 s	0.58 GFLOPs	∼1 015
RK4 (4×7.5)	∼75–80 s	0.62 GFLOPs	∼1 300
DoPri5 (bounded)	∼1 400–1 500 s	0.55 GFLOPs	∼65–115
ESRK-21 (ours)	∼130–135 s	0.81 GFLOPs	∼750

TABLE X: **NFE per epoch on Tiny-ImageNet (ΔT=30).** Fixed-step ESRK has deterministic NFE; DoPri5 incurs overhead from adaptive subdivision.

Solver	NFE/epoch	Cost
ESRK-15	600	1.0×
RK4	640	1.07×
ESRK-21	840	1.4×
DoPri5 (bounded)	11 000	18.3×

controls trainability in stiff regimes.

e) Computational trade-offs and practical guidance.: Increasing stage count introduces a predictable and linear increase in computational cost. ESRK-15 provides the best efficiency–stability trade-off in moderately stiff regimes, while ESRK-21 is preferable when gradients consistently operate near the stability boundary. When training is already stable, increasing the stage count offers diminishing returns. We observe that solver selection is regime-dependent: increasing stability-region size provides negligible benefit when dynamics are well within the stability region, but becomes critical as training approaches the stability boundary.

Practical guideline.

- $\Delta T \leq 10$: RK4 or ESRK-15 are sufficient.
- $10 < \Delta T \leq 20$: ESRK-15 offers improved stability at a modest cost.
- $\Delta T > 20$: ESRK-21 yields the most robust optimization, particularly when training exhibits instability (e.g., persistent clipping or increased spectral amplification).

Extended-stability integrators act as a form of numerical preconditioning: they improve how gradients propagate through depth without altering model expressivity. Stability-region geometry should therefore be treated as a first-class design parameter in Neural ODE training.

V. Conclusion

This work demonstrates that the stability region of the ODE solver governs the conditioning of gradient flow in Neural ODE training. Using extended-stability Runge–Kutta (ESRK) schemes, we vary the real-axis stability interval without altering model architecture, solver order, or optimization. Across CIFAR-10, CIFAR-100, and Tiny-ImageNet, and integration horizons $\Delta T \in \{10, 20, 30\}$, increasing stability-region size improves gradient-flow coherence and reduces spectral amplification, while preserving predictive accuracy. These results show that stability-region geometry acts as a numerical preconditioner, shaping gradient propagation rather than model expressivity. While evaluated on image classification tasks, these effects arise from the stability polynomial $R(h\lambda)$ and are therefore expected to extend to other Neural ODE settings, including time-series and generative models. We leave large-scale and cross-domain validation to future work. Future directions include solver–model co-design, stability-aware training mechanisms, and extending stability shaping beyond the real axis to broader continuous-time models. Overall, our findings reframe solver choice as a core design decision in continuous-depth learning, motivating numerically informed Neural ODE architectures in which stability geometry is treated as a first-class parameter.

References

[1] R. T. Q. Chen, Y. Rubanova, J. Bettencourt, and D. Duvenaud, "Neural ordinary differential equations," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2018. [Online]. Available: https://arxiv.org/abs/1806.07366

[2] S. O'Sullivan, "A class of high-order runge–kutta–chebyshev stability polynomials," *Journal of Computational Physics*, vol. 300, pp. 665–678, 2015. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0021999115005033

[3] ——, "Runge–kutta–gegenbauer explicit methods for advection-diffusion problems," *Journal of Computational Physics*, vol. 388, pp. 209–223, 2019. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0021999119301706

[4] W. Grathwohl, R. T. Chen, J. Bettencourt, I. Sutskever, and D. Duvenaud, "Ffjord: Free-form continuous dynamics for scalable reversible generative models," *arXiv preprint arXiv:1810.01367*, 2018.

[5] Y. Rubanova, R. T. Q. Chen, and D. K. Duvenaud, "Latent ODEs for irregularly-sampled time series," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2019. [Online]. Available: https://arxiv.org/abs/1907.03907

[6] B. Tzen and M. Raginsky, "Neural stochastic differential equations: Deep latent gaussian models in the diffusion limit," *arXiv preprint arXiv:1905.09883*, 2019.

[7] J. R. Dormand and P. J. Prince, "A family of embedded runge–kutta formulae," *Journal of Computational and Applied Mathematics*, vol. 6, no. 1, pp. 19–26, 1980.

[8] C. Tsitouras, "Runge–kutta pairs of order 5(4) satisfying only the first column simplifying assumption," *Computers & Mathematics with Applications*, vol. 62, no. 2, pp. 770–775, 2011.

[9] A. Ghosh, H. Behl, E. Dupont, P. Torr, and V. Namboodiri, "Steer: Simple temporal regularization for neural ode," *Advances in Neural Information Processing Systems*, vol. 33, pp. 14 831–14 843, 2020.

[10] P. Kidger, J. Morrill, J. Foster, and T. Lyons, "Neural controlled differential equations for irregular time series," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. [Online]. Available: https://arxiv.org/abs/2005.08926

[11] B. Sommeijer, L. Shampine, and J. Verwer, "Rkc: An explicit solver for parabolic pdes," *Journal of Computational and Applied Mathematics*, vol. 88, no. 2, pp. 315–326, 1998. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0377042797002197

[12] A. Abdulle, "Fourth order chebyshev methods with recurrence relation," *SIAM Journal on Scientific Computing*, vol. 23, no. 6, pp. 2041–2054, 2002.

[13] M. Torrilhon and R. Jeltsch, "Essentially optimal explicit runge–kutta methods with application to hyperbolic–parabolic equations," *Numerische Mathematik*, vol. 106, no. 3, pp. 303–334, 2007.

[14] J. H. Williamson, "Low-storage runge–kutta schemes," *Journal of Computational Physics*, vol. 35, no. 1, pp. 48–56, 1980.

[15] C. A. Kennedy, M. H. Carpenter, and R. M. Lewis, "Low-storage, explicit runge–kutta schemes for the compressible navier–stokes equations," *Applied numerical mathematics*, vol. 35, no. 3, pp. 177–219, 2000.