Fixed-Point Reasoners: Stable and Adaptive Deep Looped Transformers

Sajad Movahedi [*1 2 3] **Vera Milovanović** [*2 1 4 3] **Shlomo Libo Feigin** [*2 1 4 3] **Alexander Theus** [*4 2 1 3]
Thomas Hoffmann [4] **Valentina Boeva** [†4 5 6] **T. Konstantin Rusch** [†1 2 3 7] **Antonio Orvieto** [†1 2 3]

Abstract

Looped architectures provide an inductive bias toward learning step-by-step procedures for tasks that require compositional reasoning. The effective layers reached by looping determines the quality of the solution these models reach. Similar to deep architectures, looped architectures are prone to a signal propagation issue induced by depth as the halting decision is postponed. In this paper, we address the signal propagation issue by using pre-norm layers and residual scaling. With this architectural modification, we propose ***FPRM***: a **F**ixed-**P**oint **R**easoning **M**odel that uses fixed-point convergence as an end-to-end halting mechanism in a looped architecture. We show that fixed-point halting allows FPRM to adapt its compute to the difficulty of the task. FPRM proves effective on common reasoning benchmarks, namely Sudoku, Maze, state-tracking and ARC-AGI.

1. Introduction

Reasoning in neural networks has increasingly been framed as a problem of scaling test-time compute: a model should be able to spend more computation on inputs it finds harder [46, 54]. However, doing so requires two ingredients. **(1) flexibility:** the possibility of spending a variable amount of compute on the problem. Once the model is flexible, the next step is **(2) adaptivity:** a way to scale the compute spent on the problem; i.e., when to halt the computation.

The standard way to achieve both is through Chain-of-Thought (CoT) mechanism [60]. With CoT, the model scales compute through verbalization, and makes halting decisions based on predicting a specialized halting token.

[*]Equal contribution [†]Equal advising. [1]ELLIS Institute Tuebingen, Tuebingen, Germany [2]Max Planck Institute for Intelligent Systems, Tuebingen, Germany [3]Tuebingen AI Center, Tuebingen, Germany [4]ETH Zurich, Zurich, Switzerland [5]Swiss Institute of Bioinformatics, Lausanne, Switzerland [6]Université Paris Cité, Paris, France [7]Liquid AI. Correspondence to: Sajad Movahedi <sajad.movahedi@tue.ellis.eu>.

Proceedings of the 43rd International Conference on Machine Learning, Seoul, South Korea. PMLR 306, 2026. Copyright 2026 by the author(s).

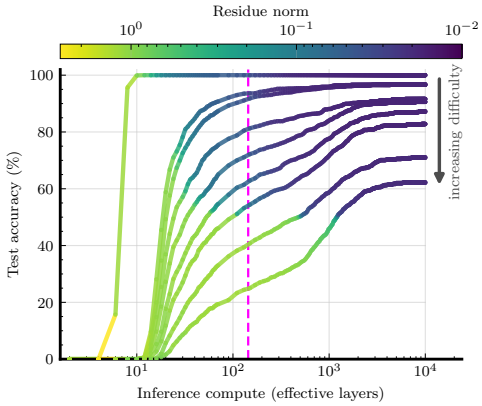


Figure 1. **FPRM allocates more compute to harder problems.** Harder inputs need more iterations before halting and peak beyond the training compute limit (dashed line); color shows residual norm.

However, the emerging behavior requires a special training regime and hand-crafted reasoning traces [31]. This makes the method complex and breaks the desirable feature of having an end-to-end training.

An alternative approach towards the end-to-end training method for reasoning models is emerging in the form of looped architectures [14, 7, 52, 32]. Whereas in CoT, the compute scales along the sequence dimension, in looped architectures it scales along the depth dimension (i):

$$\mathbf{z}_{i+1} = f_\theta(\mathbf{z}_i; \mathbf{x}), \tag{1}$$

where $f_\theta(\mathbf{z}; \mathbf{x})$ is a neural network, $\mathbf{x}$ is the input, and $\mathbf{z}_i$ is the ith latent representation. Thus, compute can be increased at test-time by looping in depth, naturally introducing flexibility from the architecture. Looped models have been shown to have an inductive bias towards learning algorithms [62, 20] and have achieved remarkable success on reasoning benchmarks. For example, Hierarchical Reasoning Model (HRM) [59] and Tiny Reasoning Model (TRM) [36] incorporate a hierarchical structure to the looping, proving effective in solving puzzle tasks such as sudoku, maze, and ARC-AGI [13].

The halting decision (i.e., deciding when to stop iterating), however, is no longer trivial in looped models. Most

approaches either fix or randomly sample the number of loops [35, 63, 25, 52, 11, 7], which eliminates adaptivity, or use external modules trained to make halting decisions [59, 36, 14]. The latter is associated with separate Adaptive Computation Time (ACT) networks [30], which introduce optimization challenges, as they require a continuous relaxation of a discrete objective. As a result, ACT can fail to provide adaptivity, which we demonstrate in the case of HRM and TRM (Figures 4, 5). We mitigate this issue by introducing a different halting mechanism: let the model loop until its hidden state converges to a fixed-point, and use the convergence itself as the halting signal. Unlike ACT, the fixed-point halting mechanism requires no external module and lets the model spend as much compute as a given input demands.

A further challenge arises as the number of loops in the model increases. This is desirable for harder tasks, but unrolling the recursions yields a large effective layers. As a result, looped architectures suffer from the same **signal propagation** issue as deep non-looped networks. To this end, we adapt architectural techniques originally developed to optimize traditional transformers at large effective layers, carefully modifying them for the Looped Transformer setting. Our key intuition is that *Looped Transformers are, in part, very deep transformers*.

Viewed this way, one design choice is surprising: looped models commonly use post-norm [14, 25, 59, 36]. Deep non-looped architectures, by contrast, prefer pre-norm, since post-norm causes unstable training [61, 45]. Yet in looped models, post-norm serves a specific purpose. It keeps the magnitude of activations bounded as the model iterates, preventing the hidden states from diverging [42] (see Figure 2). This begs the question: *can we switch the post-norm to pre-norm, while ensuring the activations stay bounded in a different way?* If so, this could make the training of looped models stable at large depths. In this paper, we use residual scaling parameters to give an affirmative answer to this question.

Our contributions are: (i) a pre-norm looped Transformer stabilized by residual scaling; (ii) fixed-point convergence as an end-to-end halting rule, with damping to suppress oscillations; and (iii) FPRM, a single-loop reasoning model that improves over hierarchical baselines on several symbolic reasoning benchmarks while adapting compute to input difficulty.

2. Background and Related Work

Looped models decouple test-time depth from parameter count by repeatedly applying a shared block, which makes them a natural architecture for algorithmic and compositional reasoning [14, 62, 20]. Recent recursive reasoning models such as HRM and TRM use hierarchical post-norm loops and learned ACT-style halting heads [59, 36]. These models are strong, but leave two questions open: how to halt adaptively without a separate learned decision module, and how to make very deep recurrent unrollings trainable.

FPRM answers these questions by combining fixed-point halting with architectural changes motivated by deep Transformers. Unlike ACT methods, which learn a discrete stopping rule through a relaxed objective [30, 6], FPRM halts when the latent dynamics converge (Figure 1). This connects to Deep Equilibrium Models and recent attractor-style reasoning models [4, 21, 34], but we use direct fixed-point iteration rather than a quasi-Newton solver, and focus on the signal-propagation bottleneck caused by large effective depth. Additional related work is provided in Section A.

3. Method

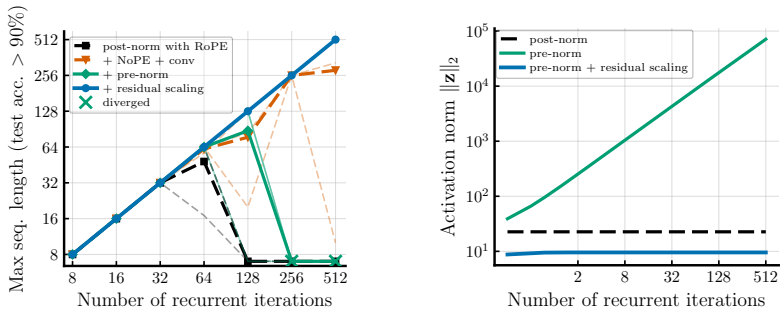
Looped architectures rely on bounded activations for stability: as the model loops, an unbounded layer can let the hidden state grow without limit (Figure 2). Current methods mostly enforce this through post-norm [36, 59, 25]. However, post-norm can restrict signal propagation at large effective depth. FPRM therefore combines three ingredients: pre-norm layers for trainability, residual scaling for bounded recurrent dynamics, and fixed-point halting for adaptive inference.

3.1. Improving signal propagation with pre-norm

We start by introducing pre-norm and post-norm in a Transformer layer. A Transformer layer consists of two sub-layers (with $f^{\ell}_{\theta^{\ell}}(.)$ denoting the $\ell^{th} \in \{1, \ldots, 2L\}$ sub-layer): multi-head attention and a feed-forward network. We denote the Looped Transformer model as defined in Equation (1), consisting of multiple Transformer layers, by $f_{\theta}(.)$. In the Transformer layer, the two sub-layers are interleaved with layer normalization (Norm). Two canonical placements of the normalization define two variants of the layer. The original *post-norm* formulation ($\mathrm{Norm}_{\mathrm{post}}$) [58] applies normalization after the residual addition with the residual stream ($\mathbf{z}^{\ell-1}$), while the *pre-norm* variant ($\mathrm{Norm}_{\mathrm{pre}}$) [61] applies normalization to the input of each sub-layer without modifying the residual stream:

$$\mathbf{z}^{\ell} = \mathrm{Norm}_{\mathrm{post}}\big(\mathbf{z}^{\ell-1} + f^{\ell}_{\theta^{\ell}}\big(\mathrm{Norm}_{\mathrm{pre}}\big(\mathbf{z}^{\ell-1}\big)\big)\big)\,, \quad \ell = 1, \ldots, 2L.$$

where L is the number of layers (Transformer blocks). In fixed-depth models, both normalization placements have been linked to training issues at large depth: post-norm bounds activation magnitudes but induces a signal propagation problem [40, 45], while pre-norm improves signal propagation but causes exponential growth in residual magnitude [40, 61]. Therefore, while using pre-norm in a deep



(a) Trainability vs. context length

(b) Activation norm at init.

Figure 2. **The blessing and the curse of depth in Looped Transformers.** A large effective layer can unlock expressivity, but also creates a stability challenge: pre-norm models without residual scaling can diverge in activation norm, while post-norm models may struggle to utilize the signal.

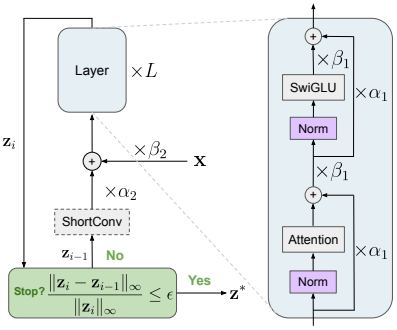


Figure 3. **FPRM architecture.** Our fixed-point Looped Transformer uses pre-norm and residual scaling for improved signal propagation.

neural network is desirable from the signal propagation stand-point, it can introduce instability due to unbounded activations.

Motivated by these observations, we investigate both normalization placements in Looped Transformers [52, 14]. In Figure 2a, we test the positive correlation between the effective layer and the expressivity (maximum sequence length with $>90\%$ test accuracy) of a Looped Transformer for the state-tracking task A_5 [44]. We observe that increasing the effective layer of a Looped Transformer with post-norm does not translate into improved expressivity, while a pre-norm variant diverges at higher depth. This divergence can be attributed to the exponential growth of the activations, apparent in Figure 2b. As the literature motivates the use of pre-norm in deep models, in the following section we try to stablize pre-norm in Looped Transformers.

3.2. Recovering boundedness via residual scaling

While pre-norm mitigates trainability problems in looped architectures, it removes the necessary boundedness condition that motivated the use of post-norm in them. This effect can be observed in Figure 2b, where the activations of the pre-norm model grow with larger effective layer, causing trainability issues observable in Figure 2a. Consequently, we propose to restore boundedness by introducing scaling parameters applied at two different scales: one over each sub-layer of the network, and one across the iterations.

Layer-wise residual scaling. Within a single application of $f_\theta(\mathbf{z};\mathbf{x})$, the residual stream and sub-layer output $f^\ell_{\theta^\ell}(\mathbf{z}^{\ell-1})$ are weighted by tied scalars (α_1, β_1) shared across all L layers, for $\ell = 1, \ldots, 2L$:

$$\mathbf{z}^\ell = \alpha_1\, \mathbf{z}^{\ell-1} + \beta_1\, f^\ell_{\theta^\ell}\big(\mathrm{Norm}_{\mathrm{pre}}(\mathbf{z}^{\ell-1})\big)\,. \quad (2)$$

Iteration-wise input mixing. Between consecutive applications of $f_\theta(\mathbf{z};\mathbf{x})$, we re-inject the input $\mathbf{x}$ with tied scalars (α_2, β_2) shared across all iterations [4]:

$$\mathbf{z}^0_{i+1} = \alpha_2\, \mathbf{z}^{2L}_i + \beta_2\, \mathbf{x}. \quad (3)$$

The two scaling schemes are not independent. With an appropriate coupling between them, the resulting recurrence is *bounded for any input*, resulting in a stable looping [47]. In the following statement, we formalize this claim:

Theorem 1 (Boundedness of FPRM iterates)**.** *Consider the model defined by Equations 2 and 3, and assume each layer map satisfies $\|f^\ell_{\theta^\ell}(\mathbf{u})\| \le c_f$ for all ℓ and input $\mathbf{u}$. Let $0 \le \alpha_1, \alpha_2 < 1$, and set*

$$\beta_2 = 1 - \alpha_2 \alpha_1^{2L}, \qquad \beta_1 = \frac{\beta_2\,(1-\alpha_1)}{1-\alpha_1^{2L}}.$$

Then the fixed-point iterates $\{\mathbf{z}^0_i\}_{i\ge 0}$ from Equation (3) are bounded, and if $\mathbf{z}^0_i \to \mathbf{z}^0_\infty$, then

$$\|\mathbf{z}^0_\infty\| \le \|\mathbf{x}\| + \alpha_2\, c_f.$$

The proof is in Section D.1.

Note that the boundedness condition of the sequence model in Theorem 1 is satisfied when using pre-norm [39]. However, boundedness still does not guarantee the looping to converge to a fixed-point, which we propose to utilize for adaptivity. In the following, we show that there exists a choice of α_2 that satisfies this requirement. Consequently, as empirically shown by [7], the model may become locally contractive during training.

Theorem 2 (Small α_2 implies convergence)**.** *Let λ_f be the Lipschitz constant of the L-layer model in Equation (2),*

i.e. $\mathbf{z}^0 \mapsto \mathbf{z}^{2L}$. *Then the looped step* $f_\theta(\,\cdot\,;\mathbf{x})$ *of Equations* (2) *and* (3) *is Lipschitz in its first argument with constant* $\alpha_2\lambda_f$. *If* $\alpha_2\lambda_f < 1$, *then* $\mathbf{z}_{i+1} = f_\theta(\mathbf{z}_i;\mathbf{x})$ *converges to a unique fixed-point* $\mathbf{z}^\star = f_\theta(\mathbf{z}^\star;\mathbf{x})$ *with* $\|f_\theta(\mathbf{z}_i;\mathbf{x}) - \mathbf{z}_i\| \le (\alpha_2\lambda_f)^i \|f_\theta(\mathbf{z}_0;\mathbf{x}) - \mathbf{z}_0\|$. *See Section D.2 for the proof.*

As in prior fixed-point models [4, 2], enforcing strong contraction by construction would limit expressivity, so we make α_1 and α_2 learnable. Appendix H.3 shows that effective initializations use small α_2 and large α_1, with learned scales spreading during training, consistent with remedies for signal propagation and rank-collapse [45, 57]. Overall, residual scaling lets the pre-norm Looped Transformer use longer looping horizons than the post-norm variant (see Figure 2).

3.3. Oscillation around the fixed-point

So far, we have been able to establish that, given a small enough α_2, the model introduced in Equation (3) becomes locally contractive and converges to a fixed-point. However, in practice the contraction factor of Theorem 2 is not itself guaranteed. For some inputs, we observe that the model often descends into an oscillatory behavior, causing the iteration to stay in a small region of latent space without converging. This non-convergent behavior is not in tension with Theorem 2, as the theorem gives a sufficient condition for convergence, not a complete characterization of the iteration's behavior. In fact, oscillation around the fixed-point can happen when the Jacobian satisfies certain conditions.

Linearizing the iteration near a fixed-point $\mathbf{z}^\star$ gives $\mathbf{z}_{i+1} - \mathbf{z}^\star \approx \mathbf{J}\,(\mathbf{z}_i - \mathbf{z}^\star)$, where $\mathbf{J} = \partial f_\theta/\partial \mathbf{z}\,|_{\mathbf{z}^\star}$. Oscillation around $\mathbf{z}^\star$ arises when $\mathbf{J}$ has an eigenvalue with $\Re(\lambda_i) < 1$ but $|\lambda_i| \ge 1$, in which case the iteration spirals around $\mathbf{z}^\star$ rather than contracting toward it. The half-plane condition $\Re(\lambda_i) < 1$ is exactly what licenses a runtime fix that does not require modifying f_θ:

Theorem 3 (Damping stabilizes oscillatory fixed–point dynamics)**.** *Suppose* $f_\theta(\,\cdot\,;\mathbf{x})$ *is continuously differentiable in a neighbourhood of a fixed-point* $\mathbf{z}^\star$, *and that every eigenvalue* λ_i *of the Jacobian* $\mathbf{J}$ *at* $\mathbf{z}^\star$ *satisfies* $\Re(\lambda_i) < 1$. *Define the damped iteration map*

$$g_{\eta,\theta}(\mathbf{z};\mathbf{x}) \;:=\; \eta\, f_\theta(\mathbf{z};\mathbf{x}) + (1-\eta)\,\mathbf{z}.$$

Then there exists $\eta_0 \in (0,1)$ *such that, for every* $\eta \in (0,\eta_0)$, *the iteration* $\mathbf{z}_{i+1} = g_{\eta,\theta}(\mathbf{z}_i;\mathbf{x})$ *converges locally to* $\mathbf{z}^\star$. *Moreover,* $g_{\eta,\theta}(\,\cdot\,;\mathbf{x})$ *and* $f_\theta(\,\cdot\,;\mathbf{x})$ *have the same fixed-points.*

The proof is in Section D.3.

Theorem 3 shows that a suitable damping factor η eliminates the oscillations while preserving the fixed-points of $f_\theta(\mathbf{z};\mathbf{x})$. We measure convergence to a fixed-point at iteration i as

$$r_i \;=\; \frac{\|\mathbf{z}_i - f_\theta(\mathbf{z}_i;\mathbf{x})\|_\infty}{\|f_\theta(\mathbf{z}_i;\mathbf{x})\|_\infty + \epsilon},$$

which serves as the halting signal: the iteration stops once r_i falls below a tolerance τ. To choose η adaptively at inference time, we use a patience mechanism that decreases η whenever this residual stops improving. We track the smallest residual observed so far, $r^\star = \min_{j\le i} r_j$. A geometric decay $\eta \leftarrow \gamma\,\eta$ with $\gamma \in (0,1)$ is applied to the step-size after P consecutive iterations with no improvement in the residues; the resulting procedure is shown in Algorithm 1.

Algorithm 1 Fixed-point optimiser FPOPT: one damped step with patience-based decay

Require: initial damping η_0; decay $\gamma \in (0,1)$; patience P
Internal state: $\eta \leftarrow \eta_0,\ p \leftarrow P,\ r^\star \leftarrow \infty$

procedure STEP($\mathbf{z}$, $\tilde{\mathbf{z}}$)
  $r \leftarrow \|\mathbf{z} - \tilde{\mathbf{z}}\|_\infty / (\|\tilde{\mathbf{z}}\|_\infty + \epsilon)$ ▷ residual
  $\mathbf{z} \leftarrow \eta\,\tilde{\mathbf{z}} + (1-\eta)\,\mathbf{z}$ ▷ damped update
  if $r < r^\star$ **then**
    $r^\star \leftarrow r,\ p \leftarrow P$ ▷ progress: reset
  else
    $p \leftarrow p - 1$
    if $p \le 0$ **and** $r > \tau$ **then**
      $\eta \leftarrow \gamma\,\eta,\ p \leftarrow P$ ▷ decay η
    end if
  end if
  return $\mathbf{z}$, r
end procedure

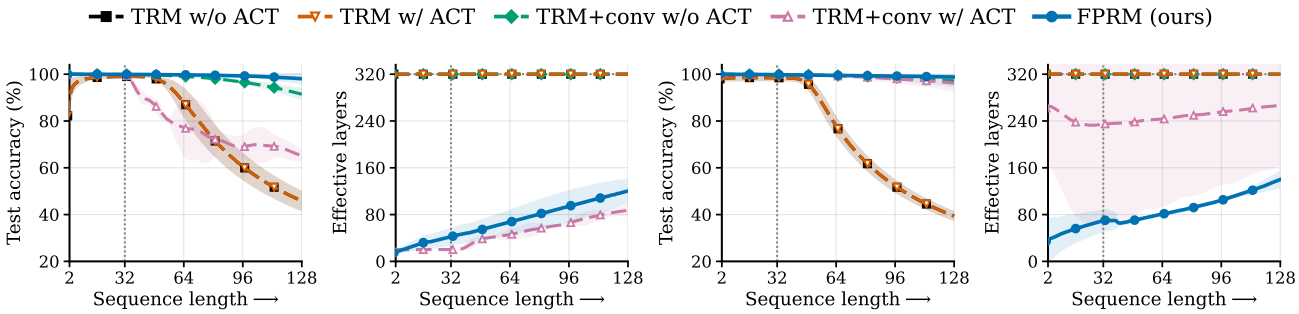
3.4. The fixed-point reasoning model

So far, we introduced modifications to improve signal propagation in Looped Transformers (Section 3.1) without sacrificing stability (Section 3.2). These modifications yield fixed-point iterations that can be made non-oscillatory (Section 3.3), making adaptivity through fixed-points reliable. We assemble these components into FPRM, illustrated in Figure 3: a single-loop Transformer that iterates until its hidden state converges, spending compute according to the input's difficulty. Since we observed no improvements from the hierarchical structure of [59], we use a classic looped architecture [14] instead.

During training, FPRM loops in truncated-BPTT windows and applies deep supervision following [59, 36]; during inference we set the window size to one and call FPOPT at each step. The optimization view, full training loop, and additional architecture details are provided in Sections B and F and Algorithm 2.

Table 1. Test accuracy on Sudoku-Extreme, Maze-Hard, ARC-AGI-1, and ARC-AGI-2. For each task, the **best overall** result is bold-face, and the best result with 7M parameters is underlined. The results denoted by † are reproduced using public checkpoints. The ARC results for URM, denoted by ‡, are only reported for Pass@1.

Model	# params.	Single Loop (No Hier.)	Sudoku-Ext. Pass@1	Maze-Hard Pass@1	ARC-1 Pass@2	ARC-2 Pass@2
Our reproduction attempt						
Attractor Model	27M	✗	71.4%	–	–	–
TRM	7M	✗	72.6%	79.0%	40.0%†	6.2%†
Reported						
HRM	27M	✗	55.0%	74.5%	40.3%	5.0%
Attractor Model	27M	✗	91.4%	**93.1%**	–	–
URM	14M	✗	77.6%	–	**≥53.8%**‡	**≥16.%**‡
TRM	7M	✗	74.7%	85.3%	44.6%	7.8%
EqR	7M	✗	93.0%	–	–	–
Attractor Model	7M	✗	54.3%	46.7%	–	–
FPRM	7M	✓	**94.2%**	87.0%	47.5%	6.2%



(a) Accuracy, A_5 *(b)* Adaptivity, A_5 *(c)* Accuracy, S_5 *(d)* Adaptivity, S_5

Figure 4. **Length generalisation and adaptive compute as a function of sequence length.** Shaded bands show 95% confidence intervals over seeds. The vertical dotted line marks the training length 32. The matched compute budget is 320 effective layers.

4. Experiments

In this section, we provide an experimental analysis of FPRM against other looped reasoning models. Our implementation builds on the public TRM codebase [36], where we also adopt the training procedure involving deep supervision. We provide the additional experimental details in Section G. We first summarize the benchmarks, then evaluate puzzle performance, adaptive compute, and depth-induced signal propagation.

4.1. Datasets

We evaluate on four families of reasoning benchmarks. Sudoku-Extreme and Maze-Hard test symbolic search on hard 9×9 Sudoku puzzles and 30×30 shortest-path mazes from [59]. ARC-1 and ARC-2 [13] test few-example grid-rule induction, reported with exact pass@2 accuracy. Finally, the A_5 and S_5 state-tracking tasks [44] test out-of-distribution length generalization for permutation composition: models train up to length 32 and are evaluated up to 128. Additional dataset details are provided in Section G.1.

4.2. Puzzle benchmarks

Table 1 compares FPRM to recent looped and recursive reasoning models on Sudoku-Extreme, Maze-Hard, ARC-1, and ARC-2. With 7M parameters and no hierarchy, FPRM is the strongest 7M model on Sudoku-Extreme, Maze-Hard, and ARC-1, and matches the public TRM checkpoint on ARC-2. On Sudoku-Extreme it also exceeds larger reported baselines, including HRM and Attractor Models, although on Maze-Hard the larger Attractor Model remains strongest. ARC-2 appears especially sensitive to model size, where larger URM and the reported TRM result outperform our 7M FPRM.

The comparison suggests that hierarchy is not necessary for strong puzzle performance: FPRM matches or improves over hierarchical post-norm baselines while using a simpler

single loop. Additional ablations, including fixed-point optimizer hyperparameters and architecture sweeps, are given in Section H.2 and Table 2.

4.3. Adaptive compute

We next test whether the halting rule allocates more computation to harder inputs. In state tracking, difficulty is sequence length; in Sudoku-Extreme, we use the number of empty cells, which correlates with puzzle difficulty [49].

State tracking. As shown in Figure 4, plain TRM and TRM+ACT fail to extrapolate to length 128, reaching $45.8\% \pm 3.9\%$ on A_5 and $39.4\% \pm 1.9\%$ on S_5. Adding causal conv1d helps length generalization, but ACT remains unreliable: on A_5, TRM+conv1d+ACT drops to $65.3\% \pm 2.2\%$, and on S_5 only some seeds learn to adapt. FPRM instead scales compute smoothly with length while maintaining high accuracy, reaching $98.1\% \pm 2.2\%$ on A_5 and $98.8\% \pm 0.9\%$ on S_5 at length 128.

Sudoku-Extreme. Figure 5 shows the same pattern when difficulty is measured by the number of empty cells: FPRM increases effective depth on harder puzzles and remains more accurate and compute-efficient than TRM with or without ACT.

4.4. Signal propagation evidence

Figure 2 isolates the depth-induced signal-propagation issue. Pre-norm alone diverges in activation norm as the number of recurrent iterations grows, while post-norm and pre-norm with residual scaling remain bounded. However, boundedness is not sufficient: only pre-norm with residual scaling converts larger iteration budgets into longer state-tracking generalization. This supports our central architectural claim that residual scaling makes pre-norm viable in looped Transformers. Additional loop-utilization and residual-scale analyses are in Sections H.1 and H.3.

5. Discussion and Conclusion

FPRM shows that latent reasoning models can obtain adaptive test-time compute without a learned halting head: the model simply iterates until its hidden state approaches a fixed point. Across state tracking and Sudoku-Extreme, this produces the desired behavior of spending more effective depth on harder inputs, while outperforming or matching hierarchical baselines of comparable size on several puzzle benchmarks. The results also suggest that part of the benefit of hierarchy in HRM/TRM-style models may be optimization-related: once signal propagation is improved directly through pre-norm and residual scaling, a single-loop architecture is sufficient for strong performance.

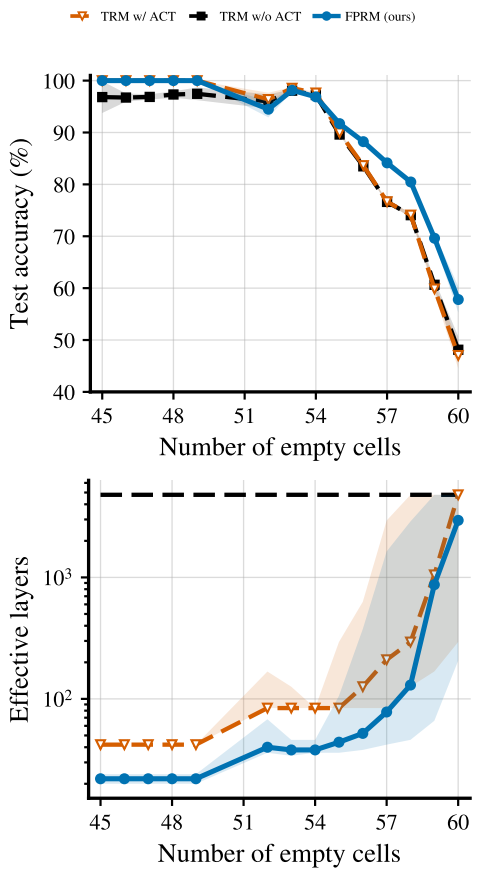


Figure 5. **FPRM adapts to Sudoku difficulty.** Top: accuracy vs. difficulty. Bottom: Median (25th–75th percentile) inference compute vs. difficulty. FPRM scales better than TRM as the number of empty cells increases.

Our study is limited to algorithmic benchmarks and Transformer backbones. Extending fixed-point halting to natural-language reasoning and to other recurrent-in-depth architectures remains open. We hope the combination of stable pre-norm looping, residual scaling, and fixed-point halting provides a useful step toward simpler and more adaptive latent reasoning models.

Acknowledgments and Disclosure of Funding

We thank Felix Sarnthein, Albert Catalan-Tatjer, Jonas Geiping, Philipp Nazari, Carl Richardson, and Nouha Dziri for the helpful discussions and comments. Alexander Theus, Vera Milovanović, and Shlomo Libo Feigin are supported by the Max Planck ETH Center for Learning Systems. Vera Milovanović and Antonio Orvieto are supported by the AI2050 program at Schmidt Sciences. Antonio Orvieto, T. Konstantin Rusch, and Sajad Movahedi acknowledge the financial support of the Hector Foundation.

References

[1] Donald G. M. Anderson. Iterative procedures for nonlinear integral equations. *J. ACM*, 12(4):547–560, 1965.

[2] Cem Anil, Ashwini Pokle, Kaiqu Liang, Johannes Treutlein, Yuhuai Wu, Shaojie Bai, J. Zico Kolter, and Roger B. Grosse. Path independent equilibrium models can better exploit test-time computation. In Sanmi Koyejo, S. Mohamed, A. Agarwal, Danielle Belgrave, K. Cho, and A. Oh, editors, *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*, 2022.

[3] Sangmin Bae, Yujin Kim, Reza Bayat, Sungnyun Kim, Jiyoun Ha, Tal Schuster, Adam Fisch, Hrayr Harutyunyan, Ziwei Ji, Aaron C. Courville, and Se-Young Yun. Mixture-of-recursions: Learning dynamic recursive depths for adaptive token-level computation. *CoRR*, abs/2507.10524, 2025.

[4] Shaojie Bai, J. Zico Kolter, and Vladlen Koltun. Deep equilibrium models. In Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily B. Fox, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 688–699, 2019.

[5] Shaojie Bai, Vladlen Koltun, and J. Zico Kolter. Stabilizing equilibrium models by jacobian regularization. In Marina Meila and Tong Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, Proceedings of Machine Learning Research, pages 554–565. PMLR, 2021.

[6] Andrea Banino, Jan Balaguer, and Charles Blundell. Pondernet: Learning to ponder. *CoRR*, abs/2107.05407, 2021.

[7] Arpit Bansal, Avi Schwarzschild, Eitan Borgnia, Zeyad Emam, Furong Huang, Micah Goldblum, and Tom Goldstein. End-to-end algorithm synthesis with recurrent networks: Extrapolation without overthinking. In Sanmi Koyejo, S. Mohamed, A. Agarwal, Danielle Belgrave, K. Cho, and A. Oh, editors, *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*, 2022.

[8] David Belanger and Andrew McCallum. Structured prediction energy networks. In Maria-Florina Balcan and Kilian Q. Weinberger, editors, *Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24, 2016*, JMLR Workshop and Conference Proceedings, pages 983–992. JMLR.org, 2016.

[9] David Belanger, Bishan Yang, and Andrew McCallum. End-to-end learning for structured prediction energy networks. In Doina Precup and Yee Whye Teh, editors, *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, Proceedings of Machine Learning Research, pages 429–439. PMLR, 2017.

[10] Hugh Blayney, Alvaro Arroyo, Johan S. Obando-Ceron, Pablo Samuel Castro, Aaron C. Courville, Michael M. Bronstein, and Xiaowen Dong. A mechanistic analysis of looped reasoning language models. *CoRR*, abs/2604.11791, 2026.

[11] Aleksandar Botev, Soham De, Samuel L. Smith, Anushan Fernando, George-Cristian Muraru, Ruba Haroun, Leonard Berrada, Razvan Pascanu, Pier Giuseppe Sessa, Robert Dadashi, Léonard Hussenot, Johan Ferret, Sertan Girgin, Olivier Bachem, Alek Andreev, Kathleen Kenealy, Thomas Mesnard, Cassidy Hardin, Surya Bhupatiraju, Shreya Pathak, Laurent Sifre, Morgane Rivière, Mihir Sanjay Kale, Juliette Love, Pouya Tafti, Armand Joulin, Noah Fiedel, Evan Senter, Yutian Chen, Srivatsan Srinivasan, Guillaume Desjardins, David Budden, Arnaud Doucet, Sharad Vikram, Adam Paszke, Trevor Gale, Sebastian Borgeaud, Charlie Chen, Andy Brock, Antonia Paterson, Jenny Brennan, Meg Risdal, Raj Gundluru, Nesh Devanathan, Paul Mooney, Nilay Chauhan, Phil Culliton, Luiz Gustavo Martins, Elisa Bandy, David Huntsperger, Glenn Cameron, Arthur Zucker, Tris Warkentin, Ludovic Peran, Minh Giang, Zoubin Ghahramani, Clément Farabet, Koray Kavukcuoglu, Demis Hassabis, Raia Hadsell, Yee Whye Teh, and Nando de Frietas. Recurrentgemma: Moving past transformers for efficient open language models. *CoRR*, abs/2404.07839, 2024.

[12] C. G. Broyden. A class of methods for solving nonlinear simultaneous equations. *Mathematics of Computation*, 19:577–593, 1965.

[13] François Chollet, Mike Knoop, Gregory Kamradt, and Bryan Landers. ARC prize 2024: Technical report, 2024.

[14] Mostafa Dehghani, Stephan Gouws, Oriol Vinyals, Jakob Uszkoreit, and Lukasz Kaiser. Universal transformers. In *7th International Conference on Learning*

Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019. OpenReview.net, 2019.

[15] Yihe Dong, Jean-Baptiste Cordonnier, and Andreas Loukas. Attention is not all you need: pure attention loses rank doubly exponentially with depth. In Marina Meila and Tong Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, Proceedings of Machine Learning Research, pages 2793–2803. PMLR, 2021.

[16] Yilun Du, Conor Durkan, Robin Strudel, Joshua B Tenenbaum, Sander Dieleman, Rob Fergus, Jascha Sohl-Dickstein, Arnaud Doucet, and Will Sussman Grathwohl. Reduce, reuse, recycle: Compositional generation with energy-based diffusion models and mcmc. In *International conference on machine learning*, pages 8489–8510. PMLR, 2023.

[17] Yilun Du, Shuang Li, Joshua B. Tenenbaum, and Igor Mordatch. Learning iterative reasoning through energy minimization. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvári, Gang Niu, and Sivan Sabato, editors, *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA*, Proceedings of Machine Learning Research, pages 5570–5582. PMLR, 2022.

[18] Yilun Du, Jiayuan Mao, and Joshua B. Tenenbaum. Learning iterative reasoning through energy diffusion, 2024.

[19] Yilun Du and Igor Mordatch. Implicit generation and modeling with energy based models. *Advances in neural information processing systems*, 32, 2019.

[20] Ying Fan, Yilun Du, Kannan Ramchandran, and Kangwook Lee. Looped transformers for length generalization. In *International Conference on Learning Representations*, volume 2025, pages 14502–14520, 2025.

[21] Jacob Fein-Ashley and Paria Rashidinejad. Solve the loop: Attractor models for language and reasoning, 2026.

[22] Samy Wu Fung, Howard Heaton, Qiuwei Li, Daniel McKenzie, Stanley J. Osher, and Wotao Yin. JFB: jacobian-free backpropagation for implicit networks. In *Thirty-Sixth AAAI Conference on Artificial Intelligence, AAAI 2022, Thirty-Fourth Conference on Innovative Applications of Artificial Intelligence, IAAI 2022, The Twelveth Symposium on Educational Advances in Artificial Intelligence, EAAI 2022 Virtual Event, February 22 - March 1, 2022*, pages 6648–6656. AAAI Press, 2022.

[23] Zitian Gao, Lynx Chen, Yihao Xiao, He Xing, Ran Tao, Haoming Luo, Joey Zhou, and Bryan Dai. Universal reasoning model. *CoRR*, abs/2512.14693, 2025.

[24] Renee Ge, Qianli Liao, and Tomaso A. Poggio. Hierarchical reasoning models: Perspectives and misconceptions. *CoRR*, abs/2510.00355, 2025.

[25] Jonas Geiping, Sean McLeish, Neel Jain, John Kirchenbauer, Siddharth Singh, Brian Bartoldson, Bhavya Kailkhura, Abhinav Bhatele, and Tom Goldstein. Scaling up test-time compute with latent reasoning: A recurrent depth approach. *Advances in Neural Information Processing Systems*, 38:41340–41391, 2026.

[26] Zhengyang Geng and J. Zico Kolter. Torchdeq: A library for deep equilibrium models, 2023.

[27] Zhengyang Geng, Xin-Yu Zhang, Shaojie Bai, Yisen Wang, and Zhouchen Lin. On training implicit models. In *NeurIPS*, pages 24247–24260, 2021.

[28] Angeliki Giannou, Shashank Rajput, Jy-yong Sohn, Kangwook Lee, Jason D. Lee, and Dimitris Papailiopoulos. Looped transformers as programmable computers. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett, editors, *International Conference on Machine Learning, ICML 2023, 23-29 July 2023, Honolulu, Hawaii, USA*, Proceedings of Machine Learning Research, pages 11398–11442. PMLR, 2023.

[29] Alexi Gladstone, Ganesh Nanduru, Md Mofijul Islam, Peixuan Han, Hyeonjeong Ha, Aman Chadha, Yilun Du, Heng Ji, Jundong Li, and Tariq Iqbal. Energy-based transformers are scalable learners and thinkers. *CoRR*, abs/2507.02092, 2025.

[30] Alex Graves. Adaptive computation time for recurrent neural networks. *CoRR*, abs/1603.08983, 2016.

[31] Daya Guo, Dejian Yang, Haowei Zhang, et al. Deepseek-r1 incentivizes reasoning in llms through reinforcement learning. *Nat.*, 645(8081):633–638, 2025.

[32] Shibo Hao, Sainbayar Sukhbaatar, DiJia Su, Xian Li, Zhiting Hu, Jason Weston, and Yuandong Tian. Training large language models to reason in a continuous latent space, 2024.

[33] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin, editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems*

2020, NeurIPS 2020, December 6-12, 2020, virtual, 2020.

[34] Benhao Huang, Zhengyang Geng, and Zico Kolter. Equilibrium reasoners: Learning attractors enables scalable reasoning, 2026.

[35] Ahmadreza Jeddi, Marco Ciccone, and Babak Taati. Loopformer: Elastic-depth looped transformers for latent reasoning via shortcut modulation. *CoRR*, abs/2602.11451, 2026.

[36] Alexia Jolicoeur-Martineau. Less is more: Recursive reasoning with tiny networks, 2025.

[37] Lukasz Kaiser and Ilya Sutskever. Neural gpus learn algorithms. In Yoshua Bengio and Yann LeCun, editors, *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*, 2016.

[38] Ferdinand Kapl, Emmanouil Angelis, Kaitlin Maile, Johannes von Oswald, and Stefan Bauer. From growing to looping: A unified view of iterative computation in llms, 2026.

[39] Hyunjik Kim, George Papamakarios, and Andriy Mnih. The lipschitz constant of self-attention. In Marina Meila and Tong Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, Proceedings of Machine Learning Research, pages 5562–5571. PMLR, 2021.

[40] Jeonghoon Kim, Byeongchan Lee, Cheonbok Park, Yeontaek Oh, Beomjun Kim, Taehwan Yoo, Seongjin Shin, Dongyoon Han, Jinwoo Shin, and Kang Min Yoo. Peri-ln: Revisiting normalization layer in the transformer architecture. In Aarti Singh, Maryam Fazel, Daniel Hsu, Simon Lacoste-Julien, Felix Berkenkamp, Tegan Maharaj, Kiri Wagstaff, and Jerry Zhu, editors, *Forty-second International Conference on Machine Learning, ICML 2025, Vancouver, BC, Canada, July 13-19, 2025*, Proceedings of Machine Learning Research. PMLR / OpenReview.net, 2025.

[41] Harsh Kohli, Srinivasan Parthasarathy, Huan Sun, and Yuekun Yao. Loop, think, & generalize: Implicit reasoning in recurrent-depth transformers. *CoRR*, abs/2604.07822, 2026.

[42] Asher Labovich. Stability and generalization in looped transformers, 2026.

[43] Yann LeCun, Sumit Chopra, Raia Hadsell, M Ranzato, Fujie Huang, et al. A tutorial on energy-based learning. *Predicting structured data*, 1(0), 2006.

[44] William Merrill, Jackson Petty, and Ashish Sabharwal. The illusion of state in state-space models. In Ruslan Salakhutdinov, Zico Kolter, Katherine A. Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp, editors, *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*, Proceedings of Machine Learning Research, pages 35492–35506. PMLR / OpenReview.net, 2024.

[45] Lorenzo Noci, Sotiris Anagnostidis, Luca Biggio, Antonio Orvieto, Sidak Pal Singh, and Aurélien Lucchi. Signal propagation in transformers: Theoretical perspectives and the role of rank collapse. In Sanmi Koyejo, S. Mohamed, A. Agarwal, Danielle Belgrave, K. Cho, and A. Oh, editors, *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*, 2022.

[46] OpenAI. Learning to reason with LLMs. https://openai.com/index/learning-to-reason-with-llms/, September 2024. OpenAI blog post, accompanying the o1 release.

[47] Antonio Orvieto, Samuel L. Smith, Albert Gu, Anushan Fernando, Çaglar Gülçehre, Razvan Pascanu, and Soham De. Resurrecting recurrent neural networks for long sequences. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett, editors, *International Conference on Machine Learning, ICML 2023, 23-29 July 2023, Honolulu, Hawaii, USA*, Proceedings of Machine Learning Research, pages 26670–26698. PMLR, 2023.

[48] Vardhan Palod, Karthik Valmeekam, Kaya Stechly, and Subbarao Kambhampati. Performative thinking? the brittle correlation between cot length and problem complexity. *CoRR*, abs/2509.07339, 2025.

[49] Marcelo O. R. Prates and Luís C. Lamb. Problem solving at the edge of chaos: Entropy, puzzles and the sudoku freezing transition. In Lefteri H. Tsoukalas, Éric Grégoire, and Miltiadis Alamaniotis, editors, *IEEE 30th International Conference on Tools with Artificial Intelligence, ICTAI 2018, 5-7 November 2018, Volos, Greece*, pages 686–693. IEEE, 2018.

[50] Zirui Ren and Ziming Liu. Are your reasoning models reasoning or guessing? A mechanistic analysis of hierarchical reasoning models, 2026.

[51] Tim Salimans and Jonathan Ho. Should ebms model the energy or the score?, 2021. Energy-Based Models Workshop, ICLR 2021.

[52] Nikunj Saunshi, Nishanth Dikkala, Zhiyuan Li, Sanjiv Kumar, and Sashank J. Reddi. Reasoning with latent thoughts: On the power of looped transformers. In *The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*. OpenReview.net, 2025.

[53] Wen-Jie Shu, Xuerui Qiu, Rui-Jie Zhu, Harold Haodong Chen, Yexin Liu, and Harry Yang. Loopvit: Scaling visual ARC with looped transformers. *CoRR*, abs/2602.02156, 2026.

[54] Charlie Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. Scaling LLM test-time compute optimally can be more effective than scaling model parameters. *CoRR*, abs/2408.03314, 2024.

[55] Shixiang Song, He Li, Zitong Wang, Boyi Zeng, Feichen Song, Yixuan Wang, Zhiqin John Xu, Ziwei He, and Zhouhan Lin. Adaponderlm: Gated pondering language models with token-wise adaptive depth. *CoRR*, abs/2603.01914, 2026.

[56] Yang Song, Jascha Sohl-Dickstein, Diederik P. Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021.

[57] Wenfang Sun, Xinyuan Song, Pengxiang Li, Lu Yin, Yefeng Zheng, and Shiwei Liu. The curse of depth in large language models. *CoRR*, abs/2502.05795, 2025.

[58] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008, 2017.

[59] Guan Wang, Jin Li, Yuhao Sun, Xing Chen, Changling Liu, Yue Wu, Meng Lu, Sen Song, and Yasin Abbasi-Yadkori. Hierarchical reasoning model. *CoRR*, abs/2506.21734, 2025.

[60] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models. In Sanmi Koyejo, S. Mohamed, A. Agarwal, Danielle Belgrave, K. Cho, and A. Oh, editors, *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*, 2022.

[61] Ruibin Xiong, Yunchang Yang, Di He, Kai Zheng, Shuxin Zheng, Chen Xing, Huishuai Zhang, Yanyan Lan, Liwei Wang, and Tie-Yan Liu. On layer normalization in the transformer architecture. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, Proceedings of Machine Learning Research, pages 10524–10533. PMLR, 2020.

[62] Liu Yang, Kangwook Lee, Robert D. Nowak, and Dimitris Papailiopoulos. Looped transformers are better at learning learning algorithms. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024.

[63] Rui-Jie Zhu, Zixuan Wang, Kai Hua, Tianyu Zhang, Ziniu Li, Haoran Que, Boyi Wei, Zixin Wen, Fan Yin, He Xing, Lu Li, Jiajun Shi, Kaijing Ma, Shanda Li, Taylor Kergan, Andrew Smith, Xingwei Qu, Mude Hui, Bohong Wu, Qiyang Min, Hongzhi Huang, Xun Zhou, Wei Ye, Jiaheng Liu, Jian Yang, Yunfeng Shi, Chenghua Lin, Enduo Zhao, Tianle Cai, Ge Zhang, Wenhao Huang, Yoshua Bengio, and Jason Eshraghian. Scaling latent reasoning via looped language models. *CoRR*, abs/2510.25741, 2025.

A. Additional Related Work

Looped models. Looped architectures address the flexibility requirement of reasoning models by decoupling the number of effective layers from parameter count, providing variable computation per input without scaling the number of parameters. This makes looping a natural inductive bias for tasks that can be solved by repeatedly applying local or compositional subroutines. Early examples include Neural GPUs [37] and Universal Transformers [14]; more recent work shows that Looped Transformers can improve length generalization, learn algorithmic structure, and approximate much deeper untied models on reasoning tasks [62, 20, 52, 38, 25, 28, 41]. Recent recursive reasoning models such as HRM [59] and TRM [36] instantiate this principle in compact architectures. Using small networks, these models have been able to outperform prominent LLM-based reasoning models. At the core of these methods is the idea of the need for a hierarchical looping mechanism, wherein the compute is distributed between a fast-looping component and a slow-looping component. URM [23] follows-up on these works and shows that by using short-conv, performance can be further improved. However, as we describe in this paper, previous works leave two central design axes open to improvement: First, *how should a looped model decide how many iterations to run on each input at test-time?* FPRM addresses this through fixed-point halting. Second, *how to make the model utilize its large effective layer?* FPRM does so by switching post-norm to pre-norm and adding residual scaling. Importantly, FPRM achieves better performance than TRM and HRM without using an hierarchical structure for the loops.

Adaptive computation. Classical ACT methods answer the adaptivity question by learning an explicit halting rule, such as halting probabilities, per-token stopping decisions, or learned distributions over computation depth [14, 6]. More recent works extend this idea by picking an adaptive depth for each token [3, 55]. Such mechanisms can, in principle, allocate more computation to harder instances, but they all add a separate learned decision process on top of the recurrent computation itself, and this process is hard to optimize because the halting decision is discrete rather than continuous. Moreover, analyses of HRM find that ACT does not always scale the inference compute with the actual difficulty of the input [50, 24], which can more generally also be the case with CoT [48] — the limitation we also observe and address by instead halting when the iteration reaches a fixed-point.

Deep equilibrium and fixed-point models. An alternative is to use the convergence of the latent dynamics as the halting criterion. In this view, computation stops when consecutive iterates become sufficiently close $\|\mathbf{z}_{i+1} - \mathbf{z}_i\| \leq \epsilon$, which corresponds to convergence toward a fixed-point $\mathbf{z}^\star = f_\theta(\mathbf{z}^\star; \mathbf{x})$. This perspective is also supported by recent mechanistic analyses of looped language models, which observe that recurrent trajectories often converge to fixed-points, suggesting that weight-tied looped models can naturally implement an implicit fixed-point computation [10]. This view is most extensively developed in Deep Equilibrium Models [4], which replace a finite stack of layers by the equilibrium point of a weight-tied transformation. However, DEQ models usually find the fixed-point via a quasi-Newton approach such Broyden's method [12, 4] or Anderson acceleration [1, 26] rather than fixed-point iterations, which makes them different from looped models. Moreover, DEQ models are difficult to optimize [5, 2, 36] , which we address by our proposed architectural changes. Concurrent to our work, Attractor Models [21] frame the iterations of TRM as a root-finding problem similar to DEQ, which they solve with Anderson acceleration. Additionally, they propose to optionally include a larger separate network to "guess" the initial latent. In another concurrent work is Equilibrium Reasoners (EqR) [34], which also adopt a fixed-point perspective and show that it is possible to scale the compute not only depth-wise, but also by making multiple initial guesses during training and inference time, favoring attractors with a wide basin. Our contributions are largely orthogonal: we target the signal-propagation issues that limit trainable depth, replacing post-norm with pre-norm and residual scaling, and damping the iteration to suppress oscillation around the fixed point. Moreover, in contrast to these works we do not use the hierarchical looping of TRM.

Score-based methods. Diffusion Models [33, 56] and Energy-Based Models (EBMs) [43] provide another way of iterative computation. An EBM learns a scalar energy function, the core idea being that the energy represents the negative log of the unnormalized density, and solutions correspond to low energy states. EBMs generate a prediction either by descending the energy landscape [8, 9] or by sampling from it with MCMC methods [19]. In comparison, diffusion models learn the score of the underlying distribution, which can be viewed as the gradient of a time-dependent energy function, thus the two are closely connected [51, 16]. While diffusion models typically integrate an SDE over a fixed time interval and in this sense are not adaptive, most relevant to our work are energy-based reasoning models [17, 18, 29], where the halting criterion is convergence to a local minimum of the energy landscape. Compared to these methods, we model the generative process via fixed-point iterations of a learned operator, as opposed to descending through a learned energy function. Learning the

operator directly enables the theoretical analysis of our architectural modifications in Section 3.

Signal propagation in looped models. As unrolled looped architectures can be viewed as deep networks, they are exposed to the same signal-propagation difficulties that arise in very deep Transformers. In deep sequence models, increasing depth can make optimization harder and can prevent later layers from being effectively used, a phenomenon often discussed as the curse of depth [15, 45, 57]. Pre-norm is the standard remedy for these issues, but in a looped setting it removes a property the architecture relies on: **bounded activations**. Bounded activations, important for stable training, are typically enforced through post-norm [25, 59, 36]. Indeed, in our own experiments a pre-norm looped model without further modification diverges in activation norm as the iteration count grows, and fails to train at the large loop counts where this growth is most severe (Figure 2). FPRM combines pre-norm with residual scaling to recover bounded and stable dynamics while still allowing gradients and representations to propagate through many iterations.

B. Optimization and Training Details

The main text describes FPRM's recurrent map and fixed-point optimizer. Here we give the optimization view that motivates truncated BPTT for fixed-point models.

B.1. Optimization of fixed-point models

One advantage of contractive fixed-point models is that they can be trained using truncated back-propagation through time (BPTT) [27]. Let $\mathbf{J} = \frac{\partial f_\theta}{\partial \mathbf{z}}(\mathbf{z}^\star; \mathbf{x})$ and $\mathbf{P} = \frac{\partial f_\theta}{\partial \theta}(\mathbf{z}^\star; \mathbf{x})$ denote the Jacobians of f_θ at the fixed-point with respect to the state and the parameters, respectively. Following the implicit function theorem we can write the gradient w.r.t. the parameters of the model as [4]:

$$\frac{d\mathbf{z}^\star}{d\theta} \;=\; (\mathbf{I} - \mathbf{J})^{-1}\,\mathbf{P}. \tag{4}$$

The Neumann series $(\mathbf{I} - \mathbf{J})^{-1} = \sum_{j \geq 0} \mathbf{J}^j$ converges, assuming $f_\theta(\mathbf{z}; \mathbf{x})$ to be contractive. Therefore, truncating the series at depth k yields the estimate:

$$\frac{d\mathbf{z}^\star}{d\theta} \;\approx\; \textstyle\sum_{j=0}^{k-1} \mathbf{J}^j\,\mathbf{P}, \tag{5}$$

which is closely related to Jacobian-free backpropagation, where the full implicit linear solve is replaced by cheaper approximate gradients [22]. The truncation depth k trades off computation against accuracy. The following proposition bounds the resulting error under contractivity.

Proposition 1 (Exponential decay of truncated-BPTT error)**.** *Let $\mathbf{z}^\star = f_\theta(\mathbf{z}^\star; \mathbf{x})$ and $\mathbf{J} = \frac{\partial f_\theta}{\partial \mathbf{z}}(\mathbf{z}^\star; \mathbf{x}) \in \mathbb{R}^{D \times D}$. If $\mathbf{J}$ is contractive in spectral norm, $\|\mathbf{J}\|_2 = \sigma < 1$, then for every $k \geq 0$,*

$$\left\| (\mathbf{I} - \mathbf{J})^{-1} - \textstyle\sum_{j=0}^{k-1} \mathbf{J}^j \right\|_F \;\leq\; \sqrt{D}\,\frac{\sigma^k}{1 - \sigma}.$$

The proof is in Section D.4. An essentially equivalent result appears in the proof of Theorem 2 of [27].

Proposition 1 allows for a fixed memory footprint during training, essentially decoupling the number of loops from the memory complexity of the model. In the same spirit, HRM [59] and TRM [36] approximate the gradient with a small number of backward passes at the fixed-point, although TRM argues that the fixed-point condition is unnecessary in practice.

C. Additional Signal-Propagation Analysis

We next isolate the role of pre-norm with residual scaling in mitigating depth-induced signal propagation problems. First, we use fixed-depth Looped Transformers to remove the confounding effect of adaptive halting. As shown in Figure 2b, pre-norm alone causes activation norms to grow rapidly with the number of recurrent iterations, while post-norm and pre-norm with residual scaling keep activations bounded. However, boundedness alone is not enough: in Figure 2a, only the pre-norm model with residual scaling turns larger iteration budgets into longer state-tracking generalization, providing evidence that the modification improves trainability at large effective depth.

We also test whether this improvement carries over to adaptive fixed-point inference. Since looped models do not have a fixed stack of layers to remove, we probe depth utilization in the opposite direction: we add recurrent computation beyond

the trained depth and measure whether performance continues to improve. Running FPRM beyond its trained depth confirms that pre-norm with residual scaling improves recurrent-depth utilization: compared to the post-norm variant, the pre-norm model benefits from additional iterations for longer while reaching comparable fixed-point residues. The full loop-utilization analysis is provided in Section H.1.

Finally, we analyze the learned residual scales in Section H.3. The main takeaway is that the best initialization keeps the residual stream dominant (large α_1) while making the loop more contractive (small α_2), consistent with both our convergence analysis and common remedies for signal-propagation problems in deep networks.

D. Proofs

D.1. Fixed-point iterations bound

Proof. We simplify our notation by omitting the pre-norm layer in Equation (2). We start by unrolling the computation across the L layers within a single fixed-point iteration. By recursively applying the layer update, we obtain

$$\begin{aligned}
\mathbf{z}_i^{2L} &= \alpha_1 \cdot \mathbf{z}_i^{2L-1} + \beta_1 \cdot f_{\theta^{2L}}^{2L}\left(\mathbf{z}_i^{2L-1}\right) && (6)\\
&= \alpha_1^2 \cdot \mathbf{z}_i^{2L-2} + \alpha_1\beta_1 \cdot f_{\theta^{2L-1}}^{2L-1}\left(\mathbf{z}_i^{2L-2}\right) + \beta_1 \cdot f_{\theta^{2L}}^{2L}\left(\mathbf{z}_i^{2L-1}\right) && (7)\\
&= \alpha_1^3 \cdot \mathbf{z}_i^{2L-3} + \alpha_1^2\beta_1 \cdot f_{\theta^{2L-2}}^{2L-2}\left(\mathbf{z}_i^{2L-3}\right) + \alpha_1\beta_1 \cdot f_{\theta^{2L-1}}^{2L-1}\left(\mathbf{z}_i^{2L-2}\right) + \beta_1 \cdot f_{\theta^{2L}}^{2L}\left(\mathbf{z}_i^{2L-1}\right) && (8)\\
&\vdots && (9)\\
&= \alpha_1^{2L} \cdot \mathbf{z}_i^0 + \beta_1 \cdot \left(\sum_{j=0}^{2L-1} \alpha_1^j \cdot f_{\theta^{2L-j}}^{2L-j}\left(\mathbf{z}_i^{2L-j-1}\right)\right). && (10)
\end{aligned}$$

Now, substituting this expression into the fixed-point update gives

$$\begin{aligned}
\mathbf{z}_{i+1}^0 &= \alpha_2 \cdot \mathbf{z}_i^{2L} + \beta_2 \cdot \mathbf{x} && (11)\\
&= \alpha_2 \cdot \left(\alpha_1^{2L} \cdot \mathbf{z}_i^0 + \beta_1 \cdot \sum_{j=0}^{2L-1} \alpha_1^j \cdot f_{\theta^{2L-j}}^{2L-j}\left(\mathbf{z}_i^{2L-j-1}\right)\right) + \beta_2 \cdot \mathbf{x} && (12)\\
&= \alpha_2\alpha_1^{2L} \cdot \mathbf{z}_i^0 + \alpha_2\beta_1 \cdot \left(\sum_{j=0}^{2L-1} \alpha_1^j \cdot f_{\theta^{2L-j}}^{2L-j}\left(\mathbf{z}_i^{2L-j-1}\right)\right) + \beta_2 \cdot \mathbf{x}. && (13)
\end{aligned}$$

For compactness, define $\rho = \alpha_2\alpha_1^{2L}$ and $\mathbf{s}_i = \sum_{j=0}^{2L-1} \alpha_1^j \cdot f_{\theta^{2L-j}}^{2L-j}\left(\mathbf{z}_i^{2L-j-1}\right)$. Then the fixed-point iteration can be written as $\mathbf{z}_{i+1}^0 = \rho \cdot \mathbf{z}_i^0 + \alpha_2\beta_1 \cdot \mathbf{s}_i + \beta_2 \cdot \mathbf{x}$. Unrolling this recursion over fixed-point iterations gives

$$\mathbf{z}_{i+1}^0 = \rho^{i+1} \cdot \mathbf{z}_0^0 + \beta_2 \left(\sum_{k=0}^{i} \rho^k\right) \cdot \mathbf{x} + \alpha_2\beta_1 \left(\sum_{k=0}^{i} \rho^k \cdot \mathbf{s}_{i-k}\right). \tag{14}$$

Since $0 \le \alpha_1, \alpha_2 < 1$, we have $0 \le \rho < 1$. Therefore, the geometric series is convergent. Taking norms and using the boundedness of each layer map, we get

$$\left\|\mathbf{z}_{i+1}^0\right\| \le \rho^{i+1} \left\|\mathbf{z}_0^0\right\| + \beta_2 \left(\sum_{k=0}^{i} \rho^k\right) \|\mathbf{x}\| + \alpha_2\beta_1 \left(\sum_{k=0}^{i} \rho^k\right) \left(\sum_{j=0}^{L-1} \alpha_1^j\right) c_f. \tag{15}$$

Letting $i \to \infty$, the first term vanishes and the two geometric sums converge, which gives

$$\limsup_{i\to\infty} \left\|\mathbf{z}_i^0\right\| \le \frac{\beta_2}{1-\rho} \|\mathbf{x}\| + \frac{\alpha_2\beta_1}{1-\rho} \left(\frac{1-\alpha_1^{2L}}{1-\alpha_1}\right) c_f. \tag{16}$$

Substituting back $\rho = \alpha_2\alpha_1^{2L}$, we obtain

$$\limsup_{i\to\infty} \left\|\mathbf{z}_i^0\right\| \le \frac{\beta_2}{1-\alpha_2\alpha_1^{2L}} \|\mathbf{x}\| + \frac{\alpha_2\beta_1\left(1-\alpha_1^{2L}\right)}{\left(1-\alpha_2\alpha_1^{2L}\right)\left(1-\alpha_1\right)} c_f. \tag{17}$$

We now set $\beta_2 = 1 - \alpha_2\alpha_1^{2L}$. This makes the coefficient of $\|\mathbf{x}\|$ equal to 1. Furthermore, setting $\beta_1 = \frac{\beta_2(1-\alpha_1)}{\left(1-\alpha_1^{2L}\right)}$ makes the coefficient of c_f equal to α_2. Therefore,

$$\limsup_{i\to\infty} \left\|\mathbf{z}_i^0\right\| \leq \|\mathbf{x}\| + \alpha_2 \cdot c_f. \tag{18}$$

In particular, if the fixed-point iteration converges to $\mathbf{z}_\infty^0$, then

$$\left\|\mathbf{z}_\infty^0\right\| \leq \|\mathbf{x}\| + \alpha_2 \cdot c_f. \tag{19}$$

This completes the proof. □

D.2. Contractive mapping

Proof. We first show that $f_\theta(.;\mathbf{x})$ is a contraction with respect to $\mathbf{z}$. For any $\mathbf{z}, \mathbf{z}'$, using the Lipschitzness of the L-layer model, we have

$$\begin{aligned}\|f_\theta(\mathbf{z};\mathbf{x}) - f_\theta(\mathbf{z}';\mathbf{x})\| &\leq \lambda_f \left\|(\alpha_2 \cdot \mathbf{z} + \beta_2 \cdot \mathbf{x}) - (\alpha_2 \cdot \mathbf{z}' + \beta_2 \cdot \mathbf{x})\right\| \\ &= \alpha_2\lambda_f \left\|\mathbf{z} - \mathbf{z}'\right\|.\end{aligned}$$

Therefore, if $0 \leq \alpha_2\lambda_f < 1$, the map $f_\theta(.;\mathbf{x})$ is strictly contractive. By the Banach fixed-point theorem, it has a unique fixed-point $\mathbf{z}^\star$, and the iteration $\mathbf{z}_{i+1} = f_\theta(\mathbf{z}_i;\mathbf{x})$ converges to $\mathbf{z}^\star$.

We now prove the residual bound. Since $\mathbf{z}_{i+1} = f_\theta(\mathbf{z}_i;\mathbf{x})$, the residual at iteration i can be written as

$$\|f_\theta(\mathbf{z}_i;\mathbf{x}) - \mathbf{z}_i\| = \|\mathbf{z}_{i+1} - \mathbf{z}_i\|.$$

Using the contraction property of $f_\theta(.;\mathbf{x})$, we get

$$\begin{aligned}\|\mathbf{z}_{i+1} - \mathbf{z}_i\| &= \|f_\theta(\mathbf{z}_i;\mathbf{x}) - f_\theta(\mathbf{z}_{i-1};\mathbf{x})\| \\ &\leq \alpha_2\lambda_f \|\mathbf{z}_i - \mathbf{z}_{i-1}\|.\end{aligned}$$

Applying this inequality recursively gives

$$\|\mathbf{z}_{i+1} - \mathbf{z}_i\| \leq (\alpha_2\lambda_f)^i \|\mathbf{z}_1 - \mathbf{z}_0\|.$$

Since $\mathbf{z}_1 = f_\theta(\mathbf{z}_0;\mathbf{x})$, we obtain

$$\|f_\theta(\mathbf{z}_i;\mathbf{x}) - \mathbf{z}_i\| \leq (\alpha_2\lambda_f)^i \|f_\theta(\mathbf{z}_0;\mathbf{x}) - \mathbf{z}_0\|.$$

This completes the proof. □

D.3. Mitigating oscillation through damping

Proof. We first show that the fixed-points of $g_{\eta,\theta}(\,\cdot\,;\mathbf{x})$ and $f_\theta(\,\cdot\,;\mathbf{x})$ coincide. Since

$$g_{\eta,\theta}(\mathbf{z};\mathbf{x}) - \mathbf{z} \;=\; \eta\big(f_\theta(\mathbf{z};\mathbf{x}) - \mathbf{z}\big),$$

and $\eta > 0$, we have $g_{\eta,\theta}(\mathbf{z};\mathbf{x}) = \mathbf{z}$ if and only if $f_\theta(\mathbf{z};\mathbf{x}) = \mathbf{z}$.

We now study the local stability of the damped iteration around $\mathbf{z}^\star$. The Jacobian of $g_{\eta,\theta}(\,\cdot\,;\mathbf{x})$ at $\mathbf{z}^\star$ is

$$\frac{\partial g_{\eta,\theta}}{\partial \mathbf{z}}(\mathbf{z}^\star;\mathbf{x}) \;=\; (1-\eta)\mathbf{I} + \eta\mathbf{J},$$

so for every eigenvalue λ_i of $\mathbf{J}$, the corresponding eigenvalue of the damped Jacobian is

$$\mu_i(\eta) \;=\; 1 - \eta + \eta\lambda_i \;=\; 1 + \eta(\lambda_i - 1).$$

Local asymptotic stability is implied by $|\mu_i(\eta)| < 1$ for all i. Writing $\lambda_i = a_i + \mathrm{i}\, b_i$ with $a_i = \Re(\lambda_i)$,

$$|\mu_i(\eta)|^2 \;=\; 1 + 2\eta(a_i - 1) + \eta^2|\lambda_i - 1|^2.$$

Hence $|\mu_i(\eta)| < 1$ if and only if $\eta|\lambda_i - 1|^2 < 2(1 - a_i)$, i.e.,

$$0 \;<\; \eta \;<\; \frac{2(1 - \Re(\lambda_i))}{|\lambda_i - 1|^2}.$$

By assumption $\Re(\lambda_i) < 1$ for every i, so each upper bound is strictly positive. Setting

$$\eta_0 \;=\; \min\left\{ 1,\; \min_i \frac{2(1 - \Re(\lambda_i))}{|\lambda_i - 1|^2} \right\} \;>\; 0,$$

we obtain $|\mu_i(\eta)| < 1$ for every i and every $\eta \in (0, \eta_0)$. Therefore $\mathbf{z}^\star$ is locally asymptotically stable under the damped iteration $\mathbf{z}_{t+1} = g_{\eta,\theta}(\mathbf{z}_t; \mathbf{x})$, and the iterates converge to $\mathbf{z}^\star$ from any sufficiently close initialization. □

D.4. Error of truncated-BPTT

Proof. Since $\|\mathbf{J}\|_2 = \sigma < 1$, the Neumann series is convergent, and we have $(\mathbf{I} - \mathbf{J})^{-1} = \sum_{j=0}^{\infty} \mathbf{J}^j$. Therefore, the error of the k-term truncated approximation is

$$\begin{aligned}\left\| (\mathbf{I} - \mathbf{J})^{-1} - \sum_{j=0}^{k-1} \mathbf{J}^j \right\|_F &= \left\| \sum_{j=k}^{\infty} \mathbf{J}^j \right\|_F \\ &\le \sum_{j=k}^{\infty} \left\| \mathbf{J}^j \right\|_F .\end{aligned}$$

Using the relation $\|\mathbf{A}\|_F \le \sqrt{D}\|\mathbf{A}\|_2$ for $\mathbf{A} \in \mathbb{R}^{D\times D}$, together with submultiplicativity of the spectral norm, we get

$$\begin{aligned}\left\| \mathbf{J}^j \right\|_F &\le \sqrt{D}\, \left\| \mathbf{J}^j \right\|_2 \\ &\le \sqrt{D}\, \|\mathbf{J}\|_2^j \\ &= \sqrt{D} \cdot \sigma^j.\end{aligned}$$

Substituting this into the previous inequality gives

$$\begin{aligned}\left\| (\mathbf{I} - \mathbf{J})^{-1} - \sum_{j=0}^{k-1} \mathbf{J}^j \right\|_F &\le \sqrt{D} \sum_{j=k}^{\infty} \sigma^j \\ &= \sqrt{D} \cdot \frac{\sigma^k}{1 - \sigma}.\end{aligned}$$

Thus, the approximation error decays as $\mathcal{O}(\sigma^k)$.

To make the corresponding gradient statement explicit, let $\mathbf{P} = \frac{\partial f_\theta}{\partial \theta}(\mathbf{z}^\star; \mathbf{x})$ and $\boldsymbol{\delta} = \frac{\partial \mathcal{L}}{\partial \mathbf{z}^\star}$. The exact implicit gradient is $\nabla_\theta \mathcal{L} = \mathbf{P}^\top \left(\mathbf{I} - \mathbf{J}\right)^{-\top} \boldsymbol{\delta}$, whereas the k-step truncated BPTT gradient is $\widehat{\nabla}_\theta^{(k)} \mathcal{L} = \mathbf{P}^\top \left(\sum_{j=0}^{k-1} \left(\mathbf{J}^\top\right)^j \right) \boldsymbol{\delta}$. Therefore,

$$\begin{aligned}\left\| \nabla_\theta \mathcal{L} - \widehat{\nabla}_\theta^{(k)} \mathcal{L} \right\|_2 &\le \|\mathbf{P}\|_2 \left\| (\mathbf{I} - \mathbf{J})^{-\top} - \sum_{j=0}^{k-1} \left(\mathbf{J}^\top\right)^j \right\|_2 \|\boldsymbol{\delta}\|_2 \\ &\le \|\mathbf{P}\|_2 \left(\sum_{j=k}^{\infty} \|\mathbf{J}\|_2^j \right) \|\boldsymbol{\delta}\|_2 \\ &= \|\mathbf{P}\|_2 \, \frac{\sigma^k}{1 - \sigma} \, \|\boldsymbol{\delta}\|_2 .\end{aligned}$$

Hence, the truncated BPTT gradient error also decays exponentially with the number of backward passes k. This completes the proof. □

E. A Toy Failure Mode for Recurrent Post-norm

In this section, we provide a toy example to showcase a failure mode of post-norm in a small setting. We take random $\mathbf{x}, \mathbf{y} \in \mathbb{R}^{n\times d}$ to be the input-output pair of sequences, with sequence length $n = 100$ and hidden-size $d = 2$, and set $\mathbf{z}_0^0 = \mathbf{x}$. Let $f_\theta(\mathbf{z}; \mathbf{x})$ to be a neural network with a single sub-layer $f^1_{\theta^1}(\mathbf{z})$ defined as:

$$f^1_{\theta^1}(\mathbf{z}) = \mathbf{z}W(\mathbf{w}),$$

where we define the rank-one map:

$$W(\mathbf{w}) = \mathbf{w}\mathbf{1}^\top \in \mathbb{R}^{2\times 2}, \qquad \mathbf{w} = (w_1, w_2)^\top,$$

with $\theta^1 = \mathbf{w}$.

The **post-norm recurrence** is defined as

$$\mathbf{z}^0_{i+1} = \mathrm{Norm}_{\mathrm{post}}\left(\mathbf{z}^0_i + f_{\theta^1}\left(\mathbf{z}^0_i\right)\right),$$

while the **pre-norm recurrence** is defined as

$$\mathbf{z}^0_{i+1} = (1-\beta)\cdot \mathbf{z}^0_i + \beta \cdot f_{\theta^1}\left(\mathrm{Norm}_{\mathrm{post}}\left(\mathbf{z}^0_i\right)\right), \qquad \beta = \frac{1}{2}.$$

Figure 6 gives a minimal version of the normalization tradeoff discussed in the main text based on this toy model. For both models we sweep a 200×200 grid over $\mathbf{w} \in \mathcal{G} = [-5, 5]^2$ and plot

$$\Delta\mathcal{L}(\mathbf{w}) = \mathcal{L}(\mathbf{w}) - \min_{\mathbf{v}\in\mathcal{G}} \mathcal{L}(\mathbf{v}),$$

where we define the loss as

$$\mathcal{L}(\mathbf{w}) = \frac{1}{nd}\left\|\mathbf{z}^0_{20}(\mathbf{w}) - \mathbf{y}\right\|_F^2,$$

i.e., at the 20th effective layer.

The figure illustrates that **boundedness does not imply trainability**. Post-norm keeps the recurrent state bounded by construction: after every step, each row is projected back onto the unit sphere. However, the same projection also removes radial information at every iteration. In this two-parameter slice, the resulting loss is organized into thin angular sectors with sharp ridges and narrow low-loss regions. Thus a random initialization of $\mathbf{w}$ is likely to start in a bounded but poorly conditioned part of the landscape, where the gradient does not point into a useful basin. This is the toy analogue of the optimization difficulty of recurrent post-norm layers.

The right panel is not bare pre-norm; it is pre-norm with residual scaling. This matters because naive pre-norm removes the projection that controls the recurrent state and can lead to activation growth (Figure 2b). With the scaled update, each row satisfies

$$\|\mathbf{z}^0_{i+1}\|_2 \le (1-\beta)\|\mathbf{z}^0_i\|_2 + \beta.$$

So the toy dynamics remain bounded while preserving a live residual stream. The broader low-loss region in Figure 6 is therefore consistent with the architecture we use in Theorem 1: pre-normalization improves signal propagation, while residual scaling replaces the boundedness mechanism that post-normalization provided.

F. Further Details About the Architecture

Fixed-point solver. Let $\mathbf{z}_i \in \mathbb{R}^{B\times T\times d}$ denote the ith latent representation (with B denoting the batch index, T the sequence index, and d the hidden size), and $\mathbf{z}_{i+1}$ the next latent representation. We index the bth batch dimension as $\mathbf{z}_i[b]$. Convergence is measured per sample $\mathbf{z}_i[b]$ by the relative L_∞ norm of the residual,

$$\mathbf{r}_i[b] = \frac{\|\mathbf{z}_{i+1}[b] - \mathbf{z}_i[b]\|_\infty}{\|\mathbf{z}_{i+1}[b]\|_\infty + \epsilon} \in \mathbb{R}.$$

A sample is declared converged when $r_i[b] < \tau$. In practice, we set τ to 0.1. But we observe that for a reasonably small choice of τ, the model is not sensitive to this value. Two safeguards bound the loop: **(1)** a hard cap on the number of iterations, and **(2)** early termination if the adaptive step size collapses below a minimum.

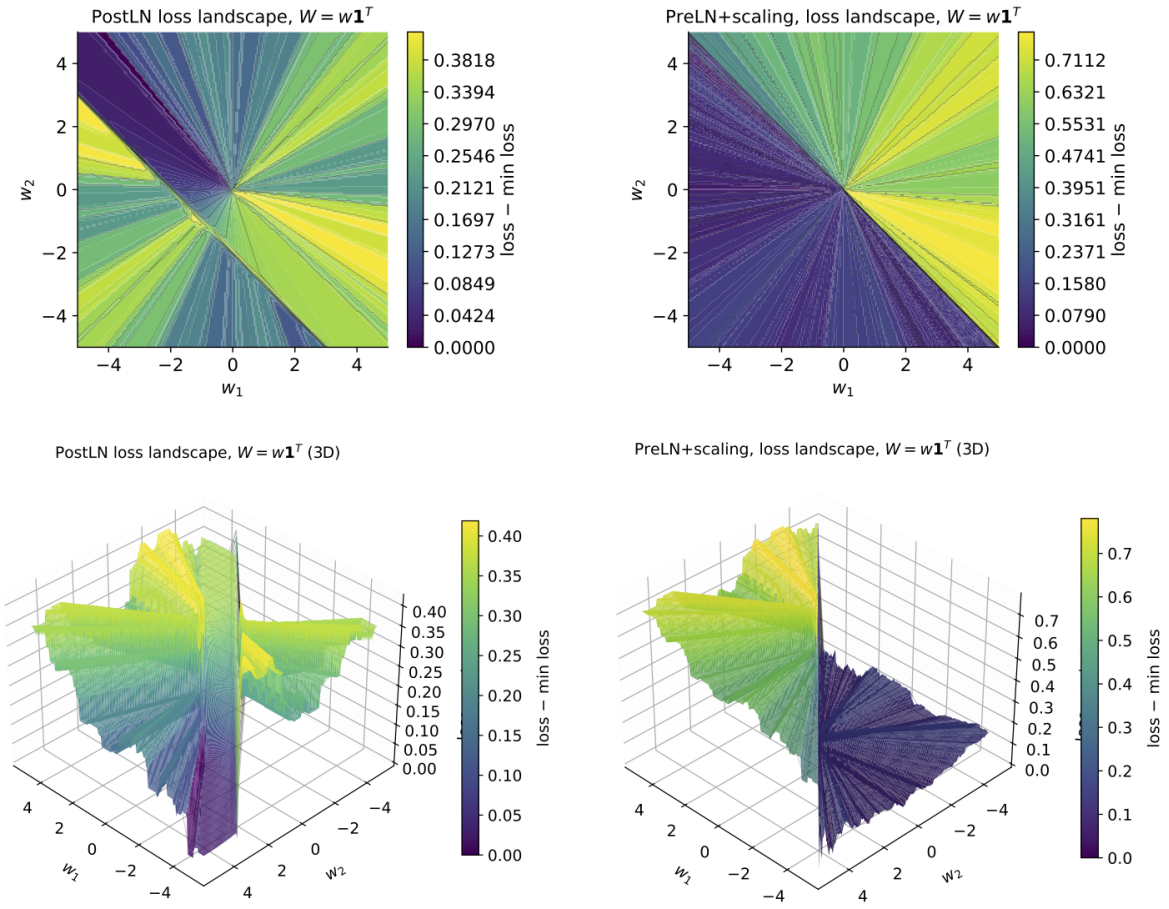


Figure 6. Landscape visualization for the setup proposed in Section E

Algorithm 2 FPRM training loop with truncated BPTT and deep supervision

Require: Model f_θ; prediction head h_ϕ; fixed-point optimiser FPOPT; model optimiser MODELOPT; input $\mathbf{x}$; target $\mathbf{y}$; BPTT depth k; initial state $\mathbf{z}_0$

```
z ← z_0
while FPOPT.CONT() do                                   ▷ outer loop
    for j = 1, ..., k do                                ▷ BPTT window
        z̃_j ← f_θ(z; x)
        z ← FPOPT.STEP(z, z̃_j)
    end for
    ŷ ← h_φ(z)                                          ▷ deep supervision
    L ← CROSSENTROPY(ŷ, y)
    MODELOPT.BACKWARD(L)
    z ← detach(z)
end while
```

Deep supervision. We adopt a similar deep supervision mechanism as HRM [59] and TRM [36]. Let T_{sup} denote the deep supervision interval. After every T_{sup} iterations, the intermediate activations of the model are decoded through the output head, the loss is computed, and truncated-BPTT is performed through the k latest iterations. Then, the computation graph is detached from the previous step. For each sequence, this process is continued until the fixed-point of the input is reached. The number of backward passes per forward pass is therefore $\lceil k/T_{\text{sup}} \rceil$. In our model, we set $T_{\text{sup}} = k$, while in TRM and HRM, T_{sup} is usually set to a larger number. However, we observe that in practice, FPRM performs a smaller number of forward and backward passes during training compared to TRM, lowering the training cost.

Depth-wise convolutions. Depth-wise convolutions have proven effective in improving the performance of looped models, at a small time and parameter complexity [53, 23]. Therefore, in FPRM we apply depth-wise convolutions on the latent representations at the beginning of each loop, which we find to be most effective. An overview of FPRM is available in Figure 3. We consider both 1D and 2D convolutions, and we find the 2D variant to be more effective at 2-dimensional tasks such as Sudoku and ARC, while the 1D variant is essential in state-tracking. However, as observed in Table 2, depth-wise convolutions seem to have a detrimental impact on the performance of TRM.

F.1. FPRM training loop

G. Additional Experimental Details

G.1. Dataset details

We provide the dataset details omitted from the main text for space.

Sudoku-Extreme. The task consists of exceptionally challenging, partially filled 9×9 Sudoku puzzles, introduced by [59]. Each sample is flattened into a sequence of length 81. The train data consists of 1000 unique samples, each augmented 1000 times, giving a total of ˜1M train samples. The test data contains $422,786$ samples. The evaluation metric is exact (sequence) accuracy.

Maze-Hard. The task consists of difficult 30×30 shortest-path maze puzzles, introduced by [59]. The train data and the test data each consist of 1000 unique samples. Following [59], we do not use augmentation for this problem. The evaluation metric is exact (sequence) accuracy.

ARC-1 and ARC-2. The Abstraction and Reasoning Corpus-1 (ARC-1) and -2 (ARC-2), introduced by [13], aim to assess the ability of the model to solve novel problems from minimal examples. ARC-1 consists of 2-3 2D grid-based input-output demonstration pairs with variable size (up to 30×30), through which the model is supposed to learn an underlying transformation rule and apply it to a held-out sample. The train data and the test data each contain ˜400 samples. ARC-2 has similar characteristics to ARC-1, but with much more challenging problems involving several complex transformations, which are less susceptible to brute-force solutions. The benchmark contains ˜1000 training samples and ˜360 test samples. The evaluation metric is exact (sequence) pass@2 (top-2 prediction) accuracy. Importantly, pretrained reasoning LLMs with

Table 2. Effect of adding FPRM's architectural modifications to TRM: pre-norm and residual scaling (α_2 only, or both α_1 and α_2), individually and in combination, evaluated with and without the conv2d layer in the TRM core. Each column reports the change in test sequence accuracy on Sudoku Extreme (%) relative to its own post-norm, no-scale baseline measured in this sweep, with the absolute accuracy shown alongside.

	w/ conv		w/o conv	
Configuration	Δ (%)	Acc. (%)	Δ (%)	Acc. (%)
Original TRM (post-norm, no residual scaling)	—	63.98	—	72.60
+ residual scaling (α_2 only)	-6.71	57.27	-4.44	68.16
+ residual scaling (α_1, α_2)	-4.47	59.51	-12.24	60.36
− post-norm + pre-norm + residual scaling (α_2 only)	-49.00	14.98	-58.87	13.73
− post-norm + pre-norm + residual scaling (α_1, α_2)	-24.52	39.46	$\mathbf{-52.83}$	19.77

CoT often struggle with these tasks. For example, Deepseek R1 (671B model) achieves 15.8% on ARC-1 and 1.3% on ARC-2, Claude 3.7 Sonnet 16K achieves 28.6% on ARC-1 and 0.7% on ARC-2.

State tracking. The A_5 and S_5 state-tracking tasks, introduced by [44], are algorithmic benchmarks based on permutation composition. Each sample consists of an initial state and a sequence of k update permutations; the model must apply the updates in order and predict the resulting final state. A_5 uses the alternating group on five elements, i.e., the subgroup of even permutations, while S_5 uses the full symmetric group on five elements. These tasks are useful proxies for stateful reasoning problems such as entity tracking, code execution, and game-state tracking, since solving them requires learning a composable update rule rather than memorizing computations at fixed lengths. We train on sequences up to $k = 32$ and evaluate out-of-distribution length generalization on sequences up to $k = 128$. The evaluation metric is exact final-state accuracy.

Weight initialization. It seems like initializing the weights using a truncated normal distribution (LeCun initialization) is common practice in looped architectures. In our experiments, it accelerates the convergence but there's very little material difference in sequence accuracy after convergence.

Grokking. There is some evidence for grokking in looped architectures, but in maze we observe convergence on the train data. And training the models for a longer period (up to 7 days) did not yield better performance.

H. Additional Empirical Analyses

H.1. Loop utilization

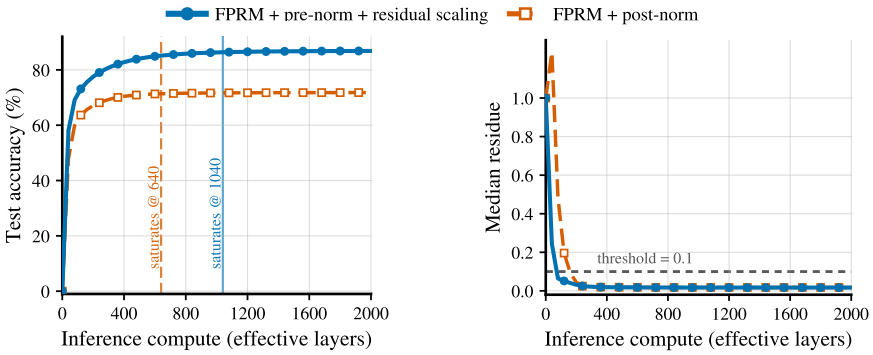
To probe whether improved signal propagation allows FPRM to make better use of additional recurrent depth, we run the model beyond its trained depth and measure where performance saturates. Figure 7 shows that the pre-norm model with residual scaling continues improving until a larger effective depth than the post-norm variant, while both exhibit comparable fixed-point residues. This suggests that the pre-norm variant does not merely converge more slowly, but converts additional iterations into more useful updates.

H.2. Fixed-point optimizer hyperparameters

We additionally study how the fixed-point optimizer controls the compute–accuracy trade-off at inference time. Figure 8 shows test accuracy and effective depth as a function of the decay rate γ for two values of the maximum patience P. Larger decay rates improve accuracy but require more effective layers before halting, whereas the choice of patience has a comparatively small effect.

H.3. Residual-scale initialization and training dynamics

While our original motivation for residual scaling was to prevent unbounded activations in pre-norm, the learned residual scales also align with common solutions to signal-propagation problems: keep the residual stream dominant and make the



(a) Test accuracy

(b) Fixed-point residues

Figure 7. **Loop-utilization of FPRM on Sudoku. (a)** Test accuracy of FPRM with pre-norm and residual scaling versus post-norm. **(b)** Median residue. The pre-norm model is better at loop utilization, while both variants have similar residues. This indicates similar latent-space convergence, with more meaningful updates in the pre-norm variant, resulting in improved performance.

Table 3. Sensitivity of FPRM to residual scaling initialization on Sudoku Extreme. Each cell reports best test sequence accuracy (%) for a given pair of initial values.

	α_2 init		
α_1 init	0.25	0.50	0.75
0.25	83.44	78.10	83.24
0.50	84.49	89.05	86.29
0.75	**94.23**	91.41	85.70

loop more contractive at initialization. We therefore perform a coarse-grained ablation over the initial values of α_1 and α_2 on Sudoku-Extreme.

Table 3 shows that the best initialization places α_1 at a high value and α_2 at a low value. The preference for large α_1 matches the intuition that the residual stream should remain dominant, while the preference for small α_2 agrees with the convergence analysis in Theorem 2. Comparing the strongest row and column also suggests that performance is slightly more sensitive to α_1 than to α_2, possibly because the gradients for α_1 come from both the MHA and MLP sub-layers.

In Figure 9, the medians of α_1 and α_2 remain close to their initial values, but the distributions widen during training. The α_1 distribution spreads much more broadly, while α_2 remains more concentrated and shifts downward for many channels. This suggests that the model learns more contractive iteration dynamics during training, in line with the observations of [7].

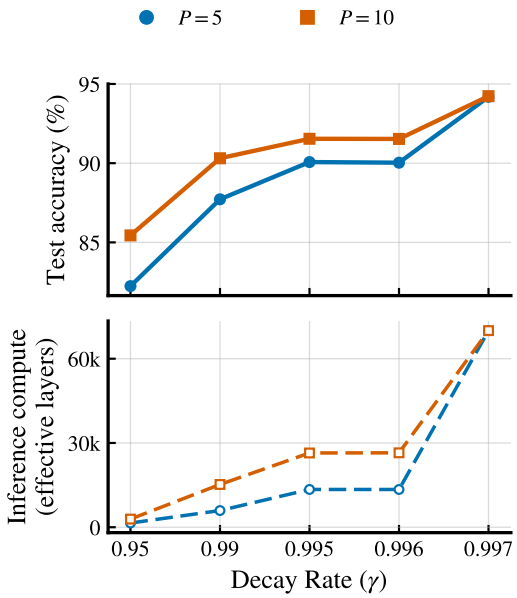


Figure 8. **Decay rate and patience.** Test accuracy and effective depth of FPRM under residue-based halting as a function of the decay rate γ, for two maximum-patience values $P \in \{5, 10\}$.

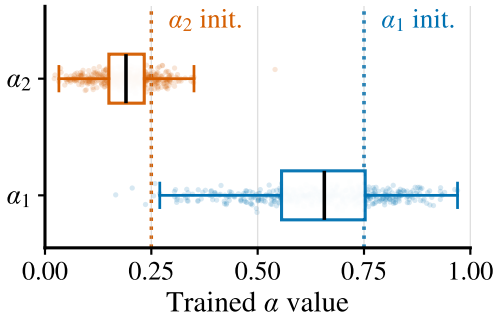


Figure 9. Distribution of the residual scales in FPRM after training on the Sudoku Extreme dataset.