SPHERICAL FIXED-POINT REASONERS: ADAPTIVE LOOP HALTING ON THE HYPERSPHERE

Anonymous authors
Paper under double-blind review

ABSTRACT

Looped (weight-tied) transformers enable adaptive computation by allocating more iterations to harder inputs. Fixed-Point Reasoners (FPRM) achieve this with a damped fixed-point solver, but operate in an unbounded Euclidean residual space with coupled rescaling and relative-norm halting. We introduce *Spherical Fixed-Point Reasoners* (SFPRM), which performs the same computation on the unit hypersphere of normalized transformers (nGPT). Input injection, damping, and halting become geodesic operations, yielding boundedness by construction and eliminating coefficient coupling. We prove fixed-point existence, local convergence under an explicit step-size condition, and global convergence under strong injection. On Sudoku-Extreme and Maze-Hard, SFPRM achieves 96.6% and 87.7%, respectively, outperforming FPRM and other baselines while allocating test-time compute more effectively. Its converged geometry is also interpretable: token–input angles capture puzzle structure and predict correctness.

1 INTRODUCTION

Scaling test-time computation has become the main lever for improving the reasoning ability of sequence models, most visibly through chain-of-thought traces (Wei et al., 2022; Wang et al., 2023; Cobbe et al., 2021; Yao et al., 2023; Snell et al., 2025). Looped transformers offer a latent alternative (Dehghani et al., 2019; Bai et al., 2019; Geiping et al., 2025): a single weight-tied block is applied repeatedly, so depth becomes an iteration count chosen at inference time, and compute can be allocated per input according to its difficulty. On algorithmic puzzles such as Sudoku, mazes, and ARC, this recipe lets small recursive reasoners compete with far larger models (Wang et al., 2025; Jolicoeur-Martineau, 2025; Movahedi et al., 2026).

A looped transformer raises two questions: *when should it stop*, and *will it stay well-behaved as it runs*? The fixed-point view, in the lineage of deep equilibrium models (Bai et al., 2019), answers the first by treating the loop as a solver for $\mathbf{z}^\star = F_\theta(\mathbf{z}^\star; \mathbf{x})$ and halt once the iterate stops moving. Fixed-Point Reasoners (FPRM) (Movahedi et al., 2026) turn this into a practical recipe: a damped iteration whose step size decays on a patience rule, a relative-norm residual as the halting test, and a boundedness theorem that keeps the recurrent state from diverging. The second question is where the Euclidean formulation shows its cost. Because the residual stream is free to grow, a boundedness theorem must couple four residual-rescaling coefficients, and the halting residual needs a normalization factor to be scale-invariant.

We introduce *Spherical Fixed-Point Reasoning Models* (SFPRM), which run FPRM's damped solver on nGPT's (Loshchilov et al., 2025) unit sphere (Figure 1). The normalized Transformer (nGPT) obtains stable dynamics by changing the *geometry* rather than adding control: every hidden state lives on the unit hypersphere $\mathbb{S}^{d-1}$ and each sub-layer is a learned angular move followed by renormalization, so the divergence FPRM must guard against cannot occur. Every control quantity of the loop becomes an angle: the input is re-injected by moving a fraction λ of the great-circle arc toward it, the damped update is a spherical linear interpolation (Slerp) whose coefficient η_i is an angular step size that FPRM's patience-based decay literally shrinks, and the halting residual is the worst-token angle between an iterate and its proposal. Our contributions are: (i) **Simplicity**: because the state cannot leave the sphere, boundedness is structural, and FPRM's coupled residual-rescaling coefficients, its boundedness theorem, and its relative-norm residual disappear rather than being re-proved; (ii) **guarantees**: a fixed point exists from continuity and input injection alone, with no

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

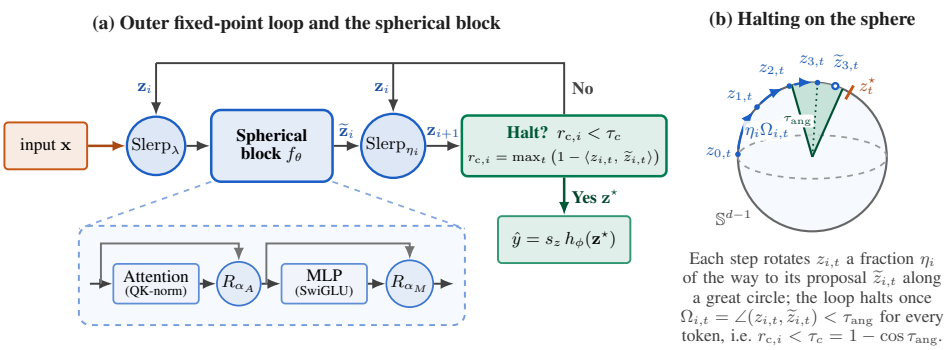


Figure 1: **SFPRM overview.** *(a)* Each iteration interpolates the current state toward the normalized input (Slerp_λ), applies the weight-tied spherical nGPT block to obtain the proposal $\widetilde{\mathbf{z}}_i$, and rotates a fraction η_i of the way toward it (Slerp_{η_i}); the loop halts once the worst-token chordal residual $r_{\text{c},i}$ falls below τ_c. The eigen-learning-rate retractions R_α inside the block keep every token state on unit sphere $\mathbb{S}^{d-1}$. *(b)* One token's iterates rotate along a great circle by shrinking angles $\eta_i \Omega_{i,t}$; the halting test compares each proposal with the current iterate, so it never needs the fixed point $z_t^\star$.

contraction or Lipschitz assumption; angular damping stabilizes any fixed point whose tangent Jacobian has real parts below one, with an explicit step-size window; and sufficiently strong injection makes it a unique global attractor; (iii) **performance**: on Sudoku-Extreme and Maze-Hard, SFPRM halts up to $3.5\times$ earlier than FPRM at equal or higher accuracy; and (iv) **interpretability**: the angle between each token state and its input anchor separates token classes during the recursion, converges to a readable encoding of the task's structure, and, on Sudoku, predicts whether the answer is correct.

2 Related Work

Looped and weight-tied transformers. Tying the weights of a transformer block (Vaswani et al., 2017) and applying it repeatedly turns depth into an iteration count at inference time (Dehghani et al., 2019; Lan et al., 2020). Looped models match much deeper non-looped ones on synthetic reasoning tasks and can simulate chain-of-thought in latent space (Saunshi et al., 2025), learn in-context algorithms more sample-efficiently (Yang et al., 2024; Gatmiry et al., 2024), emulate programmable computation (Giannou et al., 2023), generalize in length when the loop matches the algorithm's recursion (Fan et al., 2025), scale to billions of parameters (Geiping et al., 2025; Prairie et al., 2026), and supply a scaling axis for masked diffusion models (Carballo-Castro et al., 2026; Miele et al., 2026). SFPRM is a looped transformer whose loop is run as a fixed-point solve on the sphere.

Equilibrium models, adaptive computation, and halting. Adaptive Computation Time (Graves, 2016) and PonderNet (Banino et al., 2021) learn a halting distribution through an auxiliary loss, so halting is a learned head rather than a property of the dynamics. Recent methods such as HRM (Wang et al., 2025), URM (Gao et al., 2025), TRM (Jolicoeur-Martineau, 2025), and PTRM (Sghaier et al., 2026) inherit this design. Deep equilibrium models (Bai et al., 2019; 2020; Winston & Kolter, 2020; Bai et al., 2021) replace the unrolled iteration by the fixed point of a weight-tied layer and differentiate implicitly through it. Attractor models (Fein-Ashley & Rashidinejad, 2026) and equilibrium reasoners (Huang et al., 2026) make the loop's attractor explicit and halt on a solver tolerance or a residual-based selection, and FPRM (Movahedi et al., 2026) halts on a scale-invariant residual of the tied block itself, at the cost of coupled residual-rescaling coefficients that keep its Euclidean residual stream bounded; iterating past the point where the answer has settled is itself harmful (Bansal et al., 2022; Schwarzschild et al., 2021). Table 1 sorts these reasoners by whether a numerical fixed-point solve is performed and what governs halting; SFPRM alone combines a genuine solve with an intrinsic, scale-free halting criterion on a curved state space.

Normalized and hyperspherical representations. Constraining representations to a sphere is a long-standing stabilization device, for parameters (Salimans & Kingma, 2016; Miyato et al., 2018), latent variables (Davidson et al., 2018), contrastive representations (Wang & Isola, 2020), and flow language models that rotate tokens on $\mathbb{S}^{d-1}$ (Deschenaux & Gülçehre, 2026). nGPT (Loshchilov et al., 2025) carries the constraint through an entire transformer, with every hidden state and weight vector on $\mathbb{S}^{d-1}$ and each sub-layer a normalized retraction toward a learned target. SFPRM adopts that block, hence every optimizer and solver step becomes an angular update.

Table 1: Recent recurrent reasoners: HRM (Wang et al., 2025), TRM (Jolicoeur-Martineau, 2025), URM (Gao et al., 2025), Attractor (Fein-Ashley & Rashidinejad, 2026), EqR (Huang et al., 2026) and FPRM (Movahedi et al., 2026). "Appr." means no numerical fixed-point solve is performed.

Method	Recurrence	Fix. pt.	State update and input	Halting / output
HRM	Fast L-loop + slow H-loop	Appr.	Euclidean (z_L, z_H); input drives L	ACT head; read z_H
TRM	Shared net refines z, answer y	No	Euclidean; input re-read each step	ACT head; read y
URM	UT; fixed inner + ACT outer loop	No	Euclidean; short conv in MLP	ACT head, $\leq$16 steps
EqR	Tied loop; B stochastic restarts	Appr.	Euclidean; random init, path noise	ACT; lowest-residual run
Attractor	Backbone + equilibrium loop	Yes	Backbone $\tilde{y}_0$ seeds each step	Solver tol.; read $\tilde{y}^\star$
FPRM	Tied Transformer loop	Yes	(α_1, β_1) mix; (α_2, β_2) injection	Rel.-ℓ_∞ residual
SFPRM	Tied nGPT spherical loop	Yes	Eigen-LR motion; λ geodesic injection	Chordal residual

3 Spherical Fixed-Point Reasoning Models

3.1 Preliminaries

Notation. For a sequence of T tokens with hidden dimension d, the recurrent state at loop iteration i is $\mathbf{z}_i = (z_{i,1}, \ldots, z_{i,T})$ with token states $z_{i,t} \in \mathbb{R}^d$. We use i for loop iterations, t for token positions, and ℓ for sub-layers. $\mathbb{S}^{d-1} := \{z \in \mathbb{R}^d : \|z\| = 1\}$ denotes the unit hypersphere in $\mathbb{R}^d$. The embedded input is $\mathbf{x} = (x_1, \ldots, x_T)$. Tuples carry the product inner product $\langle \mathbf{u}, \mathbf{v} \rangle := \sum_{t=1}^{T} \langle u_t, v_t \rangle$ and norm $\|\mathbf{u}\|^2 := \sum_{t=1}^{T} \|u_t\|_2^2$. Lowercase f_θ always denotes the weight-tied network, uppercase $F_\theta(\cdot\,; \mathbf{x})$ the complete input-dependent recurrent map, and h_ϕ the readout.

Looped transformers and fixed-point reasoning. A looped Transformer applies a weight-tied network repeatedly to a state that is re-exposed to the same input $\mathbf{x}$ at every application: $\mathbf{z}_{i+1} = F_\theta(\mathbf{z}_i; \mathbf{x})$, for $i = 0, 1, \ldots$, and predicts with a readout $h_\phi(\mathbf{z}_i)$. A convergent iteration reaches a fixed point $\mathbf{z}^\star = F_\theta(\mathbf{z}^\star; \mathbf{x})$, the equilibrium view of Bai et al. (2019). The prediction is determined by the fixed point, while the number of iterations needed to reach it is chosen per input. FPRM (Movahedi et al., 2026) uses a pre-norm block of depth D ($2D$ attention and MLP sub-layers). Because the residual stream is free to grow indefinitely, FPRM uses tied residual coefficients (α_1, β_1) and iteration-level coefficients (α_2, β_2):

$$\mathbf{z}_i^\ell = \alpha_1 \mathbf{z}_i^{\ell-1} + \beta_1 f_{\theta^\ell}^\ell \left(\mathrm{Norm}_{\mathrm{pre}}(\mathbf{z}_i^{\ell-1})\right), \quad \ell = 1, \ldots, 2D, \qquad \mathbf{z}_{i+1}^0 = \alpha_2 \mathbf{z}_i^{2D} + \beta_2 \mathbf{x}. \tag{1}$$

Via a boundedness theorem that couples the four coefficient:,

$$\beta_2 = 1 - \alpha_2 \alpha_1^{2D}, \qquad \beta_1 = \tfrac{\beta_2(1-\alpha_1)}{1-\alpha_1^{2D}}, \qquad 0 \leq \alpha_1, \alpha_2 < 1, \tag{2}$$

ensures that the recurrent carry path (gain $\alpha_2 \alpha_1^{2D}$) is balanced against the injected input. Given a proposal $\widetilde{\mathbf{z}}_i = F_\theta(\mathbf{z}_i; \mathbf{x})$, FPRM damps linearly:

$$\mathbf{z}_{i+1} = (1 - \eta_i)\mathbf{z}_i + \eta_i \widetilde{\mathbf{z}}_i, \tag{3}$$

with a per-sample $\eta_i \in (0, 1]$ that is decayed on a patience rule whenever the residual stalls, and halts on the relative residual:

$$r_{\mathrm{E},i} = \tfrac{\|\mathbf{z}_i - \widetilde{\mathbf{z}}_i\|_\infty}{\|\widetilde{\mathbf{z}}_i\|_\infty + \epsilon}, \qquad \|\mathbf{u}\|_\infty := \max_{1 \leq t \leq T,\, 1 \leq j \leq d} |(u_t)_j|, \tag{4}$$

once $r_{\mathrm{E},i} < \tau_{\mathrm{E}}$ or the budget is exhausted. The coefficient coupling and the denominator of the residual exist to *contain* a Euclidean residual stream whose scale is not fixed. Rather than differentiate implicitly through the equilibrium as in Bai et al. (2019), FPRM trains with truncated BPTT over windows of K iterations, deep supervision, and state detachment between windows.

nGPT and spherical geometry. nGPT (Loshchilov et al., 2025) constrains every hidden state to the unit sphere $\mathbb{S}^{d-1}$ via $\mathrm{Norm}(u) = u/\|u\|_2$ for a nonzero token vector and folds each sub-layer output into the state through learned, per-coordinate *eigen learning rates* $\alpha_A, \alpha_M \in \mathbb{R}^d$:

$$\begin{aligned} \mathbf{z} &\leftarrow \mathrm{Norm}\big(\mathbf{z} + \alpha_A \odot (\mathrm{Norm}(\mathrm{Attn}(\mathbf{z})) - \mathbf{z})\big), \\ \mathbf{z} &\leftarrow \mathrm{Norm}\big(\mathbf{z} + \alpha_M \odot (\mathrm{Norm}(\mathrm{MLP}(\mathbf{z})) - \mathbf{z})\big). \end{aligned} \tag{5}$$

Norm acts token-wise on tuples, and α_A, α_M are shared across tokens. Each sub-layer is a normalized retraction toward a learned target, so unit norm is preserved by construction rather than

enforced by a constraint (Absil et al., 2008). For unit vectors u, v write $\angle(u, v) := \arccos(\langle u, v\rangle)$. For $\Omega = \angle(u, v) \in (0, \pi)$, the spherical linear interpolation:

$$\mathrm{Slerp}_\eta(u, v) = \frac{\sin((1-\eta)\Omega)}{\sin\Omega}\, u + \frac{\sin(\eta\Omega)}{\sin\Omega}\, v, \qquad \eta \in [0, 1], \tag{6}$$

moves a fraction η along the shortest great-circle arc from u to v and stays on the sphere. For coincident endpoints we set $\mathrm{Slerp}_\eta(u, u) := u$, the limit of eq. (6) as $v \to u$; at an antipodal pair ($v = -u$) the shortest arc is not unique, so we use eq. (12) (Section A.2). For small Ω, Slerp agrees with normalized linear interpolation $\mathrm{Norm}((1-\eta)u + \eta v)$ up to $O(\Omega^3)$ (Lemma 2).

3.2 SFPRM: THE RECURRENT MAP AND ITS SOLVER

SFPRM is a looped Transformer model whose recurrent state lives on the unit hypersphere and whose loop is run as a damped fixed-point solver. Algorithm 1 places it next to FPRM (Movahedi et al., 2026), its Euclidean predecessor. Every difference is a Euclidean operation replaced by its spherical one: the state is confined to $\mathcal{S}$ by a weight-tied nGPT block (line 5, eq. (5)), affine injection becomes a geodesic move a fraction λ of the arc toward the input (line 4, eq. (8)), the relative residual becomes the chordal one (line 6, eq. (9)), and linear damping becomes a rotation a fraction η_i of the way toward the proposal (line 9, eq. (10)).

State space and spherical block. SFPRM's recurrent state is a tuple of unit token vectors, $\mathbf{z}_i = (z_{i,1}, \ldots, z_{i,T}) \in \mathcal{S} := (\mathbb{S}^{d-1})^T$, where $\|z_{i,t}\|_2 = 1$ and the input $\mathbf{x} = (x_1, \ldots, x_T) \in \mathcal{S}$ is the embedded sequence normalized token-wise. The weight-tied network $f_\theta : \mathcal{S} \to \mathcal{S}$ is a stack of D nGPT layers, each an attention retraction followed by an MLP retraction as in eq. (5), with QK-normalization and eigen learning rates shared across iterations. Attention couples tokens, so f_θ acts on the product $\mathcal{S}$ rather than token by token; it is the block applied at every iteration in eq. (8). Following nGPT's logit scaling, predictions are read out as $s_z\, h_\phi(\mathbf{z})$, where, given the vocabulary $\mathcal{V}$, h_ϕ is a linear map $\mathbb{R}^d \to \mathbb{R}^{|\mathcal{V}|}$ and s_z is a learned scalar. nGPT would normalize the rows of ϕ like every other weight vector, however we keep them unnormalized so the head can set its own per-class scale while the state stays on the sphere.

Geodesic input injection. We lift Slerp from tokens to sequences by applying it position-wise. Given two unit token vectors $\mathbf{u}$ and $\mathbf{v}$, we write:

$$\mathrm{Slerp}_\eta(\mathbf{u}, \mathbf{v}) := \big(\mathrm{Slerp}_\eta(u_t, v_t)\big)_{t=1}^T, \qquad \eta \in [0, 1]. \tag{7}$$

At every iteration i, the current state is first pulled toward the input along the shortest arc using Slerp_λ of eq. (7), and the result is then passed through the nGPT block of eq. (5):

$$F_\theta(\mathbf{z}; \mathbf{x}) := f_\theta(\mathrm{Slerp}_\lambda(\mathbf{z}, \mathbf{x})), \qquad 0 < \lambda < 1. \tag{8}$$

The injection strength λ is a model parameter, fixed or learned, with the same reading in either case: it is the fraction of the arc from each token state to its input token that is traversed before the block is applied. Geodesic injection is the intrinsic form of FPRM's additive mixing $\alpha_2\mathbf{z} + \beta_2\mathbf{x}$ (eq. (1)): normalizing any positive combination $\alpha u + \beta v$ lands on the same shortest arc from $\widehat{u}$ to $\widehat{v}$, at a fraction set by the coefficients, the norms, and the angle (Lemma 3 in Section A.3). FPRM's coefficients therefore do two jobs at once: they place the state along this arc, and, through the coupling (eq. (2)), they keep its norm from drifting. On the unit sphere the second part is given by construction, and the first is done by λ alone. SFPRM thus replaces the implicit, state-dependent fraction by a single parameter and drops $(\alpha_1, \beta_1, \alpha_2, \beta_2)$ together with their coupling.

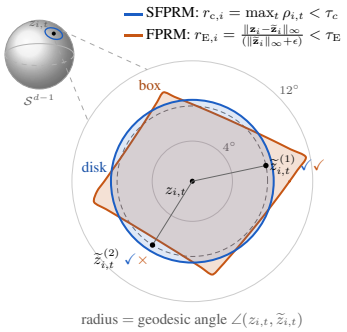


Figure 2: Proposals accepted by each halting test around a token state $z_{i,t}$. The chordal test accepts a disk; FPRM's accepts an axis-aligned ℓ_∞ section, so two proposals at the same angle can be judged differently.

Scale-free halting residual. Given the same proposal $\widetilde{\mathbf{z}}_i = F_\theta(\mathbf{z}_i; \mathbf{x})$ of eq. (8), geodesic distance on $\mathcal{S}$ is the angle between unit vectors, so SFPRM halts on the worst-token *chordal* residual, which is smooth where the angle itself is not:

$$r_{c,i} = \max_{1 \le t \le T} \rho_{i,t}, \qquad \rho_{i,t} := 1 - \langle z_{i,t}, \widetilde{z}_{i,t}\rangle = \tfrac{1}{2}\|z_{i,t} - \widetilde{z}_{i,t}\|_2^2, \tag{9}$$

and halt once $r_{c,i} < \tau_c$, where $\tau_c = 1 - \cos\tau_{\mathrm{ang}}$ for an angular tolerance τ_{ang}, so the test fires exactly when every token's proposal lies within τ_{ang} of its iterate. The *worst token* $\max_t$ is the

Algorithm 1 FPRM (left) and SFPRM (right) in one skeleton. **Black**=shared control flow; orange=FPRM-specific; blue=SFPRM-specific. Both start from a random state $\mathbf{z}_0$ and re-inject the input before every application of f_θ; the coloured lines are the only differences.

FPRM (Euclidean fixed point)

Require: Euclidean block f_θ (Transformer), head h_ϕ; tol. τ_{E}; decay γ; patience P; cap $N_{\max}$

$\mathbf{z} \leftarrow \mathbf{z}_0$ ▷ random init
$\eta \leftarrow 1$; $p \leftarrow P$; $r_{\mathrm{best}} \leftarrow \infty$
repeat
  $\widetilde{\mathbf{z}} \leftarrow \alpha_2\,\mathbf{z} + \beta_2\,\mathbf{x}$ ▷ affine input injection
  $\widetilde{\mathbf{z}} \leftarrow f_\theta(\widetilde{\mathbf{z}})$ ▷ $2D$ sub-layers, (α_1, β_1)
  $r \leftarrow \|\mathbf{z} - \widetilde{\mathbf{z}}\|_\infty / (\|\widetilde{\mathbf{z}}\|_\infty + \epsilon)$ ▷ relative-ℓ_∞
  if $r < r_{\mathrm{best}}$ **then** $r_{\mathrm{best}} \leftarrow r$; $p \leftarrow P$ **else** $p \leftarrow p-1$
  if $p \le 0$ **and** $r > \tau_{\mathrm{E}}$ **then** $\eta \leftarrow \gamma\eta$; $p \leftarrow P$
  $\mathbf{z} \leftarrow (1-\eta)\,\mathbf{z} + \eta\,\widetilde{\mathbf{z}}$ ▷ linear damping
  (train) every K: backprop $\mathrm{CE}\big(h_\phi(\mathbf{z}), y\big)$; detach $\mathbf{z}$
until $r < \tau_{\mathrm{E}}$ **or** $N_{\max}$ steps
return $h_\phi(\mathbf{z})$

SFPRM (spherical fixed point)

Require: spherical block f_θ (nGPT), head h_ϕ; tol. τ_c; decay γ; patience P; cap $N_{\max}$

$\mathbf{z} \leftarrow \mathbf{z}_0 \in \mathcal{S}$ ▷ random unit-norm init
$\eta \leftarrow 1$; $p \leftarrow P$; $r_{\mathrm{best}} \leftarrow \infty$
repeat
  $\widetilde{\mathbf{z}} \leftarrow \mathrm{Slerp}_\lambda(\mathbf{z}, \mathbf{x})$ ▷ geodesic injection
  $\widetilde{\mathbf{z}} \leftarrow f_\theta(\widetilde{\mathbf{z}})$ ▷ tied spherical block
  $r \leftarrow \max_t \big(1 - \langle z_t,\, \widetilde{z}_t \rangle\big)$ ▷ chordal residual
  if $r < r_{\mathrm{best}}$ **then** $r_{\mathrm{best}} \leftarrow r$; $p \leftarrow P$ **else** $p \leftarrow p-1$
  if $p \le 0$ **and** $r > \tau_c$ **then** $\eta \leftarrow \gamma\eta$; $p \leftarrow P$
  $\mathbf{z} \leftarrow \mathrm{Slerp}_\eta(\mathbf{z}, \widetilde{\mathbf{z}})$ ▷ geodesic damping
  (train) every K: backprop $\mathrm{CE}\big(s_z h_\phi(\mathbf{z}), y\big)$; detach $\mathbf{z}$
until $r < \tau_c$ **or** $N_{\max}$ steps
return $s_z\, h_\phi(\mathbf{z})$ ▷ scaled readout

spherical analogue of the maximum over tokens and coordinates in $r_{\mathrm{E},i}$ (eq. (4)): a few unconverged but decisive tokens should keep the iteration running. What FPRM's denominator and ϵ provide is scale invariance, and on the unit sphere that is automatic, so neither is needed. The two tests remain inequivalent: $\rho_{i,t}$ sees only the geodesic angle and so accepts a disk of radius τ_{ang} around the iterate, whereas $r_{\mathrm{E},i}$ sees the largest-magnitude coordinate and so accepts an axis-aligned ℓ_∞ region whose angular extent depends on the direction of the update, and two proposals at the same angle can get opposite verdicts (Figure 2). We adopt the chordal form for its simplicity and frame-independence.

Geodesic damping. If the residual is above tolerance, the state is advanced by a rotation toward the proposal, in place of FPRM's linear interpolation (eq. (3)):

$$\mathbf{z}_{i+1} := \mathrm{Slerp}_{\eta_i}(\mathbf{z}_i, \widetilde{\mathbf{z}}_i), \qquad 0 < \eta_i \le 1. \tag{10}$$

By Lemma 1, token t turns by exactly $\eta_i \angle(z_{i,t}, \widetilde{z}_{i,t})$, so η_i is an *angular step size*: $\eta_i = 1$ applies the raw map and smaller values shrink the rotation without leaving the sphere, so the block is always evaluated on the geometry it was trained on. The update never changes the equilibria: for $\eta_i > 0$ the step returns $\mathbf{z}_i$ if and only if $\widetilde{\mathbf{z}}_i = \mathbf{z}_i$, so the damped iteration and F_θ have exactly the same fixed points wherever the arcs are defined. As in FPRM (eq. (3)), η_i is per sample, starts at 1, and is multiplied by $\gamma < 1$ whenever the residual has not improved for P iterations. On the sphere this patience rule is literally a shrinking of the per-iteration rotation angle, a derivative-free search for a step inside the stability window of Theorem 1.

Training. Training follows FPRM: deep supervision with truncated BPTT, i.e., each optimizer step backpropagating through one segment of K geodesic steps with the state detached between segments. The residual, patience counter and step-size schedule are computed from detached states, so they steer the solver without entering the gradient; differentiating Norm projects state cotangents onto the tangent space, parameters stay Euclidean, and weight vectors are renormalized after every optimizer step. In practice, we use a guarded Slerp (eq. (12)), which replaces the sine formula by normalized linear interpolation near coincident endpoints and by an explicit arc near antipodes.

3.2.1 Theoretical guarantees

We now investigate what the geometry buys analytically. All statements concern the exact recurrence eqs. (8) to (10) with shortest-arc interpolation. Full proofs are in Appendix A.

Boundedness by construction. Because injection and damping return unit token vectors (Lemma 1), the block does so by construction (eq. (5)), and so does every branch of the guarded step (Lemma 2), the state stays on $\mathcal{S}$ for every i and every θ: recurrent magnitudes can neither explode nor vanish, which is what FPRM's boundedness theorem secures through the coefficient constraints in eq. (2). However, boundedness alone says nothing about whether a fixed point exists or is reached, which is what we address next.

Input injection guarantees a fixed point. On a sphere, continuity alone does not guarantee a fixed point since rotating a circle moves every point. Input injection repairs this by moving each token a fraction λ of the way to its input, keeping every injected state in the cap $\mathcal{C}_{\mathbf{x}} := \{\mathbf{y} \in \mathcal{S} : \angle(y_t, x_t) \leq (1-\lambda)\pi \text{ for every token } t\}$, which, unlike the sphere, can be flattened into a closed ball, where every continuous self-map has a fixed point.

Proposition 1 (Fixed-point existence)**.** *Fix* $\mathbf{x}$ *and* $0 < \lambda < 1$*. Assume* f_θ *is continuous on* $\mathcal{C}_{\mathbf{x}}$ *and* $f_\theta(\mathbf{y})_t \neq -x_t$ *for every* $\mathbf{y} \in \mathcal{C}_{\mathbf{x}}$ *and token* t*. Then the recurrent map* $F_\theta(\cdot\,; \mathbf{x})$ *in eq.* (8) *has at least one fixed point* $\mathbf{z}^\star = F_\theta(\mathbf{z}^\star; \mathbf{x})$ *in* $\mathcal{S}$*.*

The full proof (Appendix A.4) applies Brouwer's theorem (Brouwer, 1911; Ortega & Rheinboldt, 1970) to the reversed composition $H(\mathbf{y}) := \mathrm{Slerp}_\lambda(f_\theta(\mathbf{y}), \mathbf{x})$. Unlike F_θ, which injects before the block, H ends with the injection, so it maps the cap into itself, and the antipode assumption keeps it well defined and continuous; its fixed point $\mathbf{y}^\star$ gives the fixed point $\mathbf{z}^\star := f_\theta(\mathbf{y}^\star)$ of F_θ. No Lipschitz constant, contraction modulus, or spectral bound on f_θ is needed. FPRM proves existence only via global convergence, hence under contraction (Movahedi et al., 2026, Theorem 2); Brouwer alone would need the coefficient coupling that keeps a ball invariant. On the sphere compactness is structural and injection supplies the missing ball-shaped domain. Only $\mathbf{y}^\star$, however, provably lies in $\mathcal{C}_{\mathbf{x}}$, and existence implies neither uniqueness nor attraction, which the next two results address.

Angular damping stabilizes a fixed point. Let $\mathbf{z}^\star$ be a fixed point of $F_\theta(\cdot\,; \mathbf{x})$ and let $J := D_{\mathbf{z}} F_\theta(\mathbf{z}^\star; \mathbf{x})$ be its Jacobian in the recurrent state at $\mathbf{z}^\star$, with θ and $\mathbf{x}$ held fixed. Since F_θ maps $\mathcal{S}$ to itself, J sends tangent perturbations $\mathbf{u} = (u_t)_{t=1}^T$ with $\langle u_t,\, z_t^\star \rangle = 0$ to tangent perturbations, and we regard it as an endomorphism of that $T(d-1)$-dimensional tangent space. It contains all coupling between tokens through attention and need be neither orthogonal nor diagonalizable.

Theorem 1 (Local convergence of angular damping)**.** *Suppose* $F_\theta(\cdot\,; \mathbf{x})$ *is differentiable at a fixed point* $\mathbf{z}^\star$ *and every* $\zeta \in \sigma(J)$*, where* $\sigma(J)$ *is the Jacobian's set of eigenvalues, satisfies* $\mathrm{Re}(\zeta) < 1$*. Then for every fixed* $0 < \eta \leq 1$ *with*

$$\eta < \eta_{\max} := \min_{\zeta \in \sigma(J)} \frac{2\big(1 - \mathrm{Re}(\zeta)\big)}{|\zeta - 1|^2}, \tag{11}$$

the damped iteration $\mathbf{z}_{i+1} = \mathrm{Slerp}_\eta(\mathbf{z}_i, F_\theta(\mathbf{z}_i; \mathbf{x}))$ *converges to* $\mathbf{z}^\star$ *from every sufficiently close initial state, with* $\|\mathbf{z}_i - \mathbf{z}^\star\| \leq C q^i \|\mathbf{z}_0 - \mathbf{z}^\star\|$ *for some* $C > 0$ *and* $q \in (0, 1)$*. Moreover, wherever the shortest arcs are defined, the damped iteration and* F_θ *have exactly the same fixed points.*

This is the spherical counterpart of FPRM's stabilization result (Movahedi et al., 2026, Theorem 3), with an explicit window and with unit norm and fixed points preserved. The window is certified for a fixed η; the patience rule (lines 7–8 of Algorithm 1) searches for one heuristically, without guarantee. The proof (Appendix A.6) linearizes the damped step at $\mathbf{z}^\star$ to $(1-\eta)\,\mathrm{Id} + \eta J$ (Lemma 4), which moves each eigenvalue ζ of J a fraction η from 1 toward ζ; the window of eq. (11) is exactly what keeps them all inside the unit disk. A spectral radius below one then gives geometric convergence in tangent coordinates, which the coordinate map transfers to the state. Such an η exists only if $\mathrm{Re}(\zeta) < 1$ for every ζ: damping tames modes that oscillate or spiral away under the raw map, not expansive ones with $\mathrm{Re}(\zeta) \geq 1$, which training must remove (Remark 1).

Global convergence under strong injection. Finally, we identify a regime where the raw recurrence ($\eta = 1$) converges from every admissible initialization. $\mathbf{x}$ and f_θ are held fixed while λ varies.

Theorem 2 (Global convergence under strong input injection)**.** *Suppose* $f_\theta : \mathcal{S} \to \mathcal{S}$ *is continuously differentiable in a neighborhood of* $\mathbf{x}$ *and* $f_\theta(\mathbf{x})_t \neq -x_t$ *for every token* t*. Then there exists a* $\lambda_0 \in (0, 1)$*, depending on* f_θ *and* $\mathbf{x}$*, such that for every* $\lambda \in (\lambda_0, 1)$ *the map* $F_\theta(\cdot\,; \mathbf{x})$ *has a fixed point* $\mathbf{z}^\star$*, unique among all states at which the input interpolation is defined. For each such* λ *and every* $\mathbf{z}_0 \in \mathcal{S}$ *with* $z_{0,t} \neq -x_t$ *for all* t*, the iterates* $\mathbf{z}_{i+1} = F_\theta(\mathbf{z}_i; \mathbf{x})$ *are well defined and satisfy* $\|\mathbf{z}_i - \mathbf{z}^\star\| \leq C q^{i-1}$ *for all* $i \geq 1$*, with* $C > 0$ *and* $q \in (0, 1)$ *independent of* $\mathbf{z}_0$*.*

The proof (Appendix A.7, with an explicit λ_0) uses logarithmic coordinates at $\mathbf{x}$, where injection multiplies by $1 - \lambda$: for λ near one, the first injection pulls every admissible state into a small ball around $\mathbf{x}$ on which the map is a Banach contraction, so only that ball matters. FPRM's global result (Movahedi et al., 2026, Theorem 2) instead rests on a *global* Lipschitz bound and a small residual coefficient, which the sphere lacks: injection does not contract near the input's antipode. Together the three results cover every injection strength: a fixed point exists for all λ, strong injection makes it a unique global attractor, and damping gives local convergence under weaker injection.

Table 2: **Comparison with published recurrent reasoners.** Reported numbers are taken from the respective papers. Tf = Transformer, Mixer = MLP-Mixer, rec. = recurrent refinement, FP = fixed-point solve. Within the 5–7M group the best result is bold and the second best underlined.

	Model	Architecture	# params.	Single Loop (No Hier.)	Sudoku-Ext. Pass@1	Maze-Hard Pass@1	ARC-1 Pass@2	ARC-2 Pass@2
Larger	HRM	rec. Tf	27M	✗	55.0%	74.5%	40.3%	5.0%
	Attractor Model	Tf + FP loop	27M	✗	91.4%	93.1%	—	—
	URM	rec. Tf + conv	14M	✗	77.6%	—	—	—
5–7M	TRM	rec. Tf	7M	✗	74.7%	85.3%	44.6%	**7.8%**
	EqR	rec. Mixer + restarts	5-7M	✗	93.0%	—	—	—
	Attractor Model	FP Tf	7M	✗	54.3%	46.7%	—	—
	FPRM	FP Tf	7M	✓	<u>94.2%</u>	<u>87.0%</u>	**47.5%**	<u>6.2%</u>
	SFPRM (ours)	spherical FP Tf	7M	✓	**96.6%**	**87.7%**	<u>47.4%</u>	3.8%

4 Empirical Evaluation

With our experiments we answer three research questions: **(RQ1)** *Accuracy.* How does SFPRM compare to FPRM and other recurrent reasoners baselines? Does constraining the recurrence on the sphere cost accuracy? (Section 4.1) **(RQ2)** *Convergence and halting.* How effectively does SFPRM utilize the learned recurrence under the angular halting criterion and is the fixed-point unique? (Section 4.2) **(RQ3)** *Geometry of the computation.* What does the fixed–point iteration write into the state, and does the spherical parameterization make it readable? (Section 4.3)

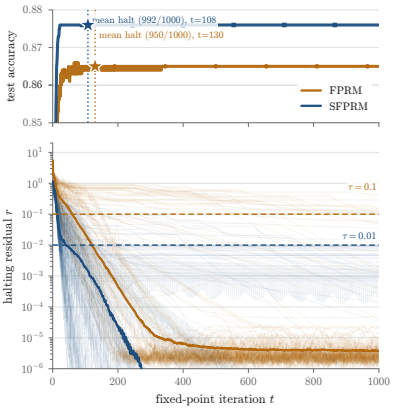


Figure 3: **Exact accuracy and halting on Maze-Hard** (Sudoku: fig. 9). SFPRM vs. FPRM evaluation on 1000 test puzzles. *Top:* exact accuracy; mean halt and number of examples halting. *Bottom:* per-sequence residual (thin), median (heavy)).

Setup. We evaluate SFPRM on puzzle benchmarks: Maze-Hard, Sudoku-Extreme (Wang et al., 2025), ARC-1 and ARC-2 (Chollet et al., 2024), used to assess generalization when trained with limited number of examples. SFPRM and FPRM share a 6.83M-parameter Transformer backbone and follow the hyperparameters of Movahedi et al. (2026) (details in Appendix B). SFPRM trains its weight matrices with Muon (Jordan et al., 2024) and its embeddings, normalization scales, and scalars with AdamW, regularizes with variational and weight dropout, and learns the injection fraction λ, except on Sudoku (λ fixed at 0.1). It halts once $r_{c,i} < \tau_c = 0.01$ (eq. (9)), i.e. once every token's lies within 8.1° of its iterate.

4.1 RQ1: Accuracy on puzzle benchmarks

Table 2 compares SFPRM with published recurrent reasoners. On Sudoku-Extreme, SFPRM reaches 96.6% exact accuracy on all 422,786 test puzzles, 2.4 points above FPRM. Because SFPRM memorizes the standard build of 1k base puzzles augmented 1000×, we train it on 5k puzzles augmented 200×: the same number of training examples, but five times as many distinct puzzles. On Maze-Hard, it reaches 87.7% on the 1,000 test mazes, above TRM's and FPRM's published results [1] and behind only the 27M-parameter Attractor model (Fein-Ashley & Rashidinejad, 2026). On ARC-1, it matches FPRM, while on ARC-2 the sphere costs accuracy, though all models score low.

Answer to RQ1. Confining the state to the sphere does not cost accuracy: SFPRM outperforms FPRM on Sudoku-Extreme and Maze-Hard and matches it on ARC-1.

4.2 RQ2: Convergence and halting

Loop utilization and halting. To see how the two solvers converge and when they stop, we run each trained model with halting disabled for $N_{\max}$ iterations, decode every iterate $\mathbf{z}_i$, and record the

[1] Our own FPRM runs fell short of the published 94.2% and 87.0% (Movahedi et al., 2026), so we compare against the published numbers and use the released checkpoints for evaluations of FPRM.

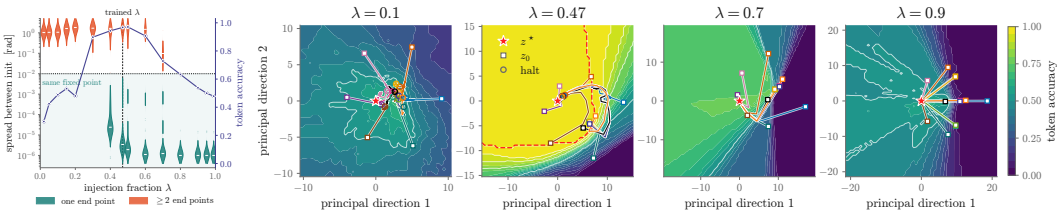


Figure 4: **Global convergence under input injection (SFPRM, Sudoku). (a)** Largest angle between the end states of eight initializations, over 32 puzzles, against λ (teal: all coincide, coral: ≥ 2 end states); indigo, right axis: token accuracy. **(b)** Trajectories from the eight $\mathbf{z}_0$ (squares) to $\mathbf{z}^\star$ (star) in their top-2 principal plane at four values of λ, coloured by the token accuracy of the decoded state; circles mark the first iteration with $r_{c,i} < \tau_c$, dashed red the exact-match region.

residual its solver tests. Halting is counted per sequence, i.e., a sequence halts once its residual stays below the threshold (Appendix C.4). Figure 3 shows the result on Maze and Figure 9 on Sudoku. Both residuals fall quickly, and both models saturate their accuracy within 20–35 iterations, so every later loop is spent after the answer stops changing. SFPRM converges faster and stops sooner: it halts on $992/1000$ mazes against FPRM's 950, and on average 22 iterations earlier (108 vs. 130).

Fixed-point uniqueness. Theorem 2 predicts that for $\lambda > \lambda_0$ the raw recurrence $\mathbf{z}_{i+1} = F_\theta(\mathbf{z}_i; \mathbf{x})$ contracts to one fixed point from every initialization. We test this by overriding λ at inference in the trained model (learned $\lambda = 0.47$) and running the raw map, without damping, in fp32 for 400 steps on 32 evenly spaced test puzzles, from eight initializations each: six random states, the input $\mathbf{x}$, and its antipode $-\mathbf{x}$, which the theorem excludes. The largest geodesic angle between the eight end states measures uniqueness without a reference point (Figure 4a). For $\lambda \geq 0.8$ all eight end states coincide on every puzzle. At the trained $\lambda = 0.47$ the raw map reaches an attracting fixed point on 91% of the puzzles, and on the puzzle of Figure 4b all eight paths end inside the exact-match region. At $\lambda = 0.1$ the raw iteration does not converge, while for $\lambda \geq 0.7$ it converges within a few steps to a unique fixed point that no longer solves the puzzle.

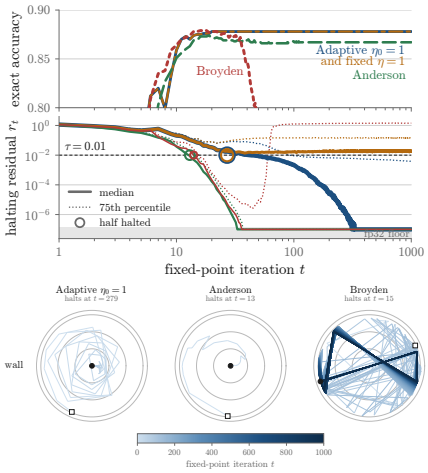


Figure 5: **Fixed-point solvers** (SFPRM, Maze). **(top)** Accuracy, halting disabled. **(bottom)** A path token trajectory: radius is the geodesic angle (log-scaled) to $\mathbf{z}^\star$, rings $0.1/1/10/100°$.

Varying the fixed-point solver. At inference we replace the damped step of Algorithm 1 by two standard solvers, keeping the model and the halting rule: Anderson acceleration (Anderson, 1965; Walker & Ni, 2011) and limited-memory Broyden (Broyden, 1965), the solver of deep equilibrium models (Bai et al., 2019). Both cost one evaluation of F_θ per iteration, like the damped step (details in Appendix C.5). For both models Anderson halts earlier than the default solver, on every Maze example, while staying within one point of its accuracy (Table 5 and Figure 5). Broyden halts about as early on Maze but diverges afterwards, so it relies on stopping in time. Without damping ($\eta = 1$), 37–100% of examples never reach τ, and on Sudoku accuracy drops as well. We leave training with these solvers, rather than swapping them in at inference, for future work.

Answer to RQ2. SFPRM uses its loop iterations more effectively than FPRM: its angular criterion halts earlier on Sudoku-Extreme and Maze-Hard at equal or higher accuracy. Under strong input injection the raw iteration converges to a unique fixed point, as Theorem 2 predicts.

4.3 RQ3: THE GEOMETRY OF THE COMPUTATION

The state reveals the task structure. On the sphere a token state's only free quantity is its direction, so we probe the computation through the *anchor angle* $\angle(z_{i,t}, x_t)$ between each cell's iterate and its unit-normalized input embedding, the quantity that injection acts on (eq. (8)).

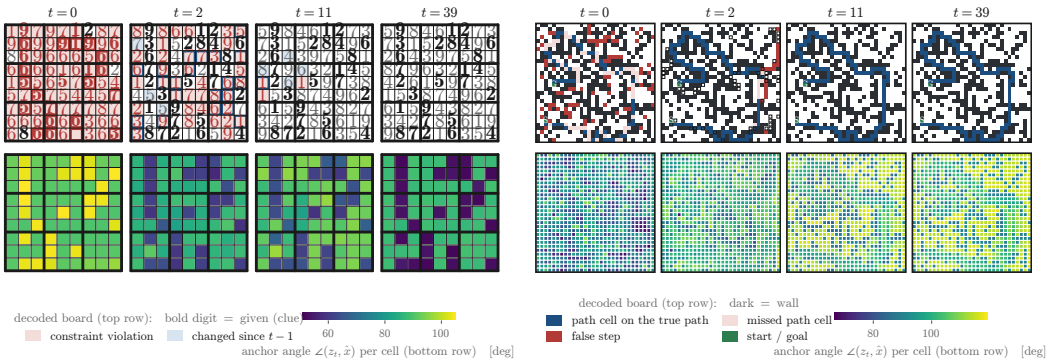


Figure 6: **SFPRM rollout frames** (FPRM in Figures 20 and 21). *Top row*: the board decoded from $\mathbf{z}_i$; *bottom row*: each cell's angle to its own input anchor. *left*: Sudoku-Extreme; *right*: Maze-Hard.

Figure 6 shows four frames of one rollout per task (FPRM and more puzzles in Figures 20 and 21), and Figures 7 and 13 show the angle over loops for every cell of a test puzzle, clustered by cell class. On Sudoku, we can see that, for SFPRM, the clue tokens rotate toward their anchors while the blanks end near orthogonal. By i=39 the anchor map is a binary image of the clue mask. Each loop iteration writes *the cell class information* into the direction of the state. On the other hand, FPRM encodes the mask mainly in the norm of its state (Figure 8), a scale set by training rather than by the geometry. Its anchor map shows only small differences between clues and blanks. Similarly, on Maze, SFPRM settles walls at ≈90°, empty cells at ≈113°, and path cells at ≈102°. FPRM separates walls from free cells but not path from empty cells.

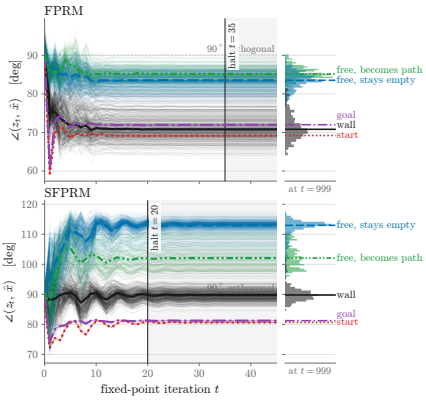


Figure 7: **Anchor angles by cell class on Maze**: cells (thin lines), class means (heavy), the halt iteration (vertical line), and the final distribution (margin).

The final angle predicts correctness. On unsolved Sudoku puzzles both SFPRM cell classes end farther from their anchors (Figure 8): the clues at ≈65° instead of ≈52°, almost disjoint from the solved ones, and the blanks at ≈92° instead of ≈88°. The converged angle thus flags a wrong answer without labels, shared by every cell and puzzle. FPRM's final norms separate the two outcomes far less: its clue and blank norms overlap widely.

Answer to RQ3. SFPRM's loop writes task structure into the state, and on the sphere it is legible as an angle to the input: the clue mask on Sudoku, the path on Maze, and, on Sudoku, whether the answer is correct. FPRM's angles and state norm carry this structure only faintly or partially.

5 Conclusion

We introduced SFPRM, which runs FPRM's damped fixed-point loop on nGPT's hypersphere. This change of geometry assures norm boundedness by construction and eliminates the need for FPRM's coupled residual-rescaling coefficients. We provide theoretical guarantees under weaker assumptions than its Euclidean counterpart. Empirically, on puzzle and reasoning tasks, SFPRM halts more effectively at equal or higher accuracy than FPRM. Finally, we showed that the angle each token keeps to its input turns out to be a readable encoding of the task's structure and a label-free predictor of correctness. Natural next steps are to scale SFPRM to language modeling and to train it with the Anderson and Broyden solvers, which we have so far used only at inference.

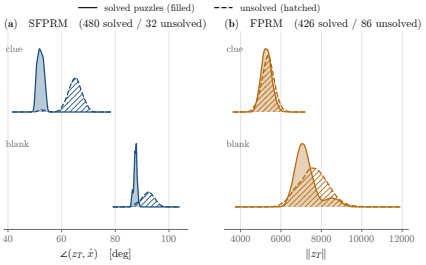


Figure 8: **Final anchor angle vs. correctness** on 512 Sudoku puzzles: angle (SFPRM) and state norm (FPRM) of clue and blank cells, for solved (filled) and unsolved (hatched) puzzles.

Reproducibility Statement

The experimental protocol is documented in Appendix B and Table 3 provides a list of the used hyperparameters in our experiments. The inference-time solver description and comparison from Section 4.2 is provided in Appendix C.5. We implemented SFPRM on top of the public FPRM code base (Movahedi et al., 2026). The code is provided as supplementary material. All runs were trained using Nvidia A100 GPUs for a single seed each.

AI Use Statement

The central idea of running FPRM's damped fixed-point solver on the unit hypersphere of nGPT, with geodesic input injection, angular damping, and a chordal halting residual, originated with the authors. For the theoretical analysis, the authors provided proof strategies and detailed guidance, while AI tools assisted with developing, writing, and checking parts of the proofs. The authors subsequently checked, corrected, and edited these proofs. The implementation of SFPRM on top of the FPRM code base was written with assistance from Claude Opus 5 and verified by the authors. The experimental design, analyses, and the figures were produced by the authors with LLM assistance. Claude was used to restructure and edit the manuscript for length and readability, including condensing the main text to the page limit. The authors reviewed all AI-assisted work and take responsibility for the final content.

References

P.-A. Absil, Robert Mahony, and Rodolphe Sepulchre. *Optimization Algorithms on Matrix Manifolds*. Princeton University Press, 2008.

Donald G. Anderson. Iterative procedures for nonlinear integral equations. *Journal of the ACM*, 12 (4):547–560, 1965.

Shaojie Bai, J. Zico Kolter, and Vladlen Koltun. Deep equilibrium models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

Shaojie Bai, Vladlen Koltun, and J. Zico Kolter. Multiscale deep equilibrium models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

Shaojie Bai, Vladlen Koltun, and J. Zico Kolter. Stabilizing equilibrium models by Jacobian regularization. In *International Conference on Machine Learning (ICML)*, 2021.

Andrea Banino, Jan Balaguer, and Charles Blundell. PonderNet: Learning to ponder. In *ICML Workshop on Automated Machine Learning (AutoML)*, 2021.

Arpit Bansal, Avi Schwarzschild, Eitan Borgnia, Zeyad Emam, Furong Huang, Micah Goldblum, and Tom Goldstein. End-to-end algorithm synthesis with recurrent networks: Logical extrapolation without overthinking. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

L. E. J. Brouwer. Über Abbildung von Mannigfaltigkeiten. *Mathematische Annalen*, 71:97–115, 1911.

Charles G. Broyden. A class of methods for solving nonlinear simultaneous equations. *Mathematics of Computation*, 19(92):577–593, 1965.

Alba Carballo-Castro, Julianna Piskorz, Paulius Rauba, Mihaela van der Schaar, and Pascal Frossard. Recursive scaling in masked diffusion models. *arXiv preprint arXiv:2606.18022*, 2026.

François Chollet, Mike Knoop, Gregory Kamradt, and Bryan Landers. ARC prize 2024: Technical report. *arXiv preprint arXiv:2412.04604*, 2024.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Łukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

Tim R. Davidson, Luca Falorsi, Nicola De Cao, Thomas Kipf, and Jakub M. Tomczak. Hyperspherical variational auto-encoders. In *Conference on Uncertainty in Artificial Intelligence (UAI)*, 2018.

Mostafa Dehghani, Stephan Gouws, Oriol Vinyals, Jakob Uszkoreit, and Łukasz Kaiser. Universal transformers. In *International Conference on Learning Representations (ICLR)*, 2019.

Justin Deschenaux and Çağlar Gülçehre. Language modeling with hyperspherical flows. *arXiv preprint arXiv:2605.11125*, 2026.

Ying Fan, Yilun Du, Kannan Ramchandran, and Kangwook Lee. Looped transformers for length generalization. In *International Conference on Learning Representations (ICLR)*, 2025.

Jacob Fein-Ashley and Paria Rashidinejad. Solve the loop: Attractor models for language and reasoning. In *ICML 2026 Workshop on Foundations of Deep Generative Models: Understanding Memorization, Generalization, and Reasoning*, 2026.

Zitian Gao, Lynx Chen, Yihao Xiao, He Xing, Ran Tao, Haoming Luo, Joey Zhou, and Bryan Dai. Universal reasoning model. *arXiv preprint arXiv:2512.14693*, 2025.

Khashayar Gatmiry, Nikunj Saunshi, Sashank J. Reddi, Stefanie Jegelka, and Sanjiv Kumar. Can looped transformers learn to implement multi-step gradient descent for in-context learning? In *International Conference on Machine Learning (ICML)*, 2024.

Jonas Geiping, Sean Michael McLeish, Neel Jain, John Kirchenbauer, Siddharth Singh, Brian R. Bartoldson, Bhavya Kailkhura, Abhinav Bhatele, and Tom Goldstein. Scaling up test-time compute with latent reasoning: A recurrent depth approach. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025.

Angeliki Giannou, Shashank Rajput, Jy-yong Sohn, Kangwook Lee, Jason D. Lee, and Dimitris Papailiopoulos. Looped transformers as programmable computers. In *International Conference on Machine Learning (ICML)*, pp. 11398–11442, 2023.

Alex Graves. Adaptive computation time for recurrent neural networks. *arXiv preprint arXiv:1603.08983*, 2016.

Benhao Huang, Zhengyang Geng, and J. Zico Kolter. Equilibrium reasoners: Learning attractors enables scalable reasoning. In *International Conference on Machine Learning (ICML)*, 2026.

Alexia Jolicoeur-Martineau. Less is more: Recursive reasoning with tiny networks. *arXiv preprint arXiv:2510.04871*, 2025.

Keller Jordan, Yuchen Jin, Vlado Boza, Jiacheng You, Franz Cesista, Laker Newhouse, and Jeremy Bernstein. Muon: An optimizer for hidden layers in neural networks. Blog post, 2024. URL `https://kellerjordan.github.io/posts/muon/`.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. ALBERT: A lite BERT for self-supervised learning of language representations. In *International Conference on Learning Representations (ICLR)*, 2020.

Ilya Loshchilov, Cheng-Ping Hsieh, Simeng Sun, and Boris Ginsburg. nGPT: Normalized transformer with representation learning on the hypersphere. In *International Conference on Learning Representations (ICLR)*, 2025.

Andrea Miele, Yiming Qin, Alba Carballo-Castro, Justin Deschenaux, and Pascal Frossard. Fixed-point masked generative modeling. *arXiv preprint arXiv:2605.31215*, 2026.

Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral normalization for generative adversarial networks. In *International Conference on Learning Representations (ICLR)*, 2018.

Sajad Movahedi, Vera Milovanović, Shlomo Libo Feigin, Alexander Theus, Thomas Hofmann, Valentina Boeva, T. Konstantin Rusch, and Antonio Orvieto. Fixed-point reasoners: Stable and adaptive deep looped transformers. *arXiv preprint arXiv:2606.18206*, 2026.

James M. Ortega and Werner C. Rheinboldt. *Iterative Solution of Nonlinear Equations in Several Variables*. Academic Press, New York, 1970.

Thomas Pethick, Wanyun Xie, Kimon Antonakopoulos, Zhenyu Zhu, Antonio Silveti-Falls, and Volkan Cevher. Training deep learning models with norm-constrained LMOs. In *International Conference on Machine Learning (ICML)*, 2025.

Hayden Prairie, Zachary Novack, Taylor Berg-Kirkpatrick, and Daniel Y. Fu. Parcae: Scaling laws for stable looped language models. *arXiv preprint arXiv:2604.12946*, 2026.

Tim Salimans and Diederik P. Kingma. Weight normalization: A simple reparameterization to accelerate training of deep neural networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2016.

Nikunj Saunshi, Nishanth Dikkala, Zhiyuan Li, Sanjiv Kumar, and Sashank J. Reddi. Reasoning with latent thoughts: On the power of looped transformers. In *International Conference on Learning Representations (ICLR)*, 2025.

Avi Schwarzschild, Eitan Borgnia, Arjun Gupta, Furong Huang, Uzi Vishkin, Micah Goldblum, and Tom Goldstein. Can you learn an algorithm? Generalizing from easy to hard problems with recurrent networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

Amin Sghaier, Ali Parviz, and Alexia Jolicoeur-Martineau. Probabilistic tiny recursive model. *arXiv preprint arXiv:2605.19943*, 2026.

Charlie Victor Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. Scaling LLM test-time compute optimally can be more effective than scaling parameters for reasoning. In *International Conference on Learning Representations (ICLR)*, 2025.

Ayhan Suleymanzade, Chanhyuk Lee, Floor Eijkelboom, Nicholas M. Boffi, İsmail İlkan Ceylan, and Jinwoo Kim. Thinking with looped flows. *arXiv preprint arXiv:2609.11801*, 2026.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.

Homer F. Walker and Peng Ni. Anderson acceleration for fixed-point iterations. *SIAM Journal on Numerical Analysis*, 49(4):1715–1735, 2011.

Guan Wang, Jin Li, Yuhao Sun, Xing Chen, Changling Liu, Yue Wu, Meng Lu, Sen Song, and Yasin Abbasi Yadkori. Hierarchical reasoning model. *arXiv preprint arXiv:2506.21734*, 2025.

Tongzhou Wang and Phillip Isola. Understanding contrastive representation learning through alignment and uniformity on the hypersphere. In *International Conference on Machine Learning (ICML)*, 2020.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc V. Le, Ed H. Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. In *International Conference on Learning Representations (ICLR)*, 2023.

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 *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

Ezra Winston and J. Zico Kolter. Monotone operator equilibrium networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

Liu Yang, Kangwook Lee, Robert D. Nowak, and Dimitris Papailiopoulos. Looped transformers are better at learning learning algorithms. In *International Conference on Learning Representations (ICLR)*, 2024.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Thomas L. Griffiths, Yuan Cao, and Karthik Narasimhan. Tree of thoughts: Deliberate problem solving with large language models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

A PROOFS FOR THE SFPRM GUARANTEES

Throughout this appendix, the already token-wise normalized input $\mathbf{x} \in \mathcal{S}$ is fixed and

$$\mathcal{S} = (\mathbb{S}^{d-1})^T, \qquad \angle(u, v) := \arccos(\langle u, v\rangle), \qquad u, v \in \mathbb{S}^{d-1}.$$

The integer T is the number of tokens. Every spherical operation is applied independently to the T token vectors. In particular, we use the sequence-level convention from eq. (7). All operands passed to Norm are assumed nonzero. Exact Slerp means interpolation along the unique shortest arc, so its endpoints are assumed non-antipodal. Numerical guards require separate care because an arbitrary antipodal fallback need not be continuous.

A.1 BASIC GEOMETRY OF SPHERICAL INTERPOLATION

Lemma 1 (Geometry of SLERP). *Let $u, v \in \mathbb{S}^{d-1}$ with $v \neq -u$, and let $s \in [0, 1]$. Then*

$$\begin{aligned} \|\mathrm{Slerp}_s(u, v)\|_2 &= 1, \\ \angle(u, \mathrm{Slerp}_s(u, v)) &= s\,\angle(u, v), \\ \angle(\mathrm{Slerp}_s(u, v), v) &= (1 - s)\angle(u, v). \end{aligned}$$

Moreover, if $\Omega := \angle(u, v) > 0$, then

$$\mathrm{Slerp}_s(u, v) = \cos(s\Omega)\, u + \sin(s\Omega)\, e, \qquad e := \frac{v - \cos\Omega\, u}{\sin\Omega},$$

where e is a unit vector orthogonal to u.

Proof. Set $\Omega = \angle(u, v)$. If $\Omega = 0$, then $v = u$ and $\mathrm{Slerp}_s(u, u) = u$ by definition. Its norm is one and both angles are zero, proving all three claims in this case. For $0 < \Omega < \pi$, define

$$e := \frac{v - \cos\Omega\, u}{\sin\Omega}.$$

Then $\langle u, e\rangle = 0$, $\|e\|_2 = 1$, and $v = \cos\Omega\, u + \sin\Omega\, e$. Substituting this identity into the SLERP formula and using the sine addition rule gives

$$\mathrm{Slerp}_s(u, v) = \cos(s\Omega)u + \sin(s\Omega)e.$$

The two terms are orthogonal, so the result has unit norm. The same expression places it at angle $s\Omega$ from u and $(1 - s)\Omega$ from v. □

A.2 GUARDED INTERPOLATION

The implementation replaces exact SLERP by a guarded step. The sine formula of eq. (6) gives $0/0$ at $\Omega = 0$ and is ill conditioned nearby, especially in bf16, although $\mathrm{Slerp}_\eta(u, u) = u$; at $\Omega = \pi$ the shortest arc is not unique. Write $c = \langle u, v\rangle$ and $\mathrm{nlerp}_\eta(u, v) := \mathrm{Norm}\big((1 - \eta)u + \eta v\big)$, and near an antipode fix an explicit arc through a unit vector $e(u, v) \perp u$, namely $e(u, v) = \mathrm{Norm}(P_u v)$ with $P_u = I - uu^\top$ when $\|P_u v\|_2 > \epsilon_v$ and a fixed unit vector $e_0(u) \perp u$ otherwise. The guarded interpolation is

$$\mathrm{gSlerp}_\eta(u, v) = \begin{cases} \mathrm{nlerp}_\eta(u, v), & c > 1 - \epsilon_\Omega, \\ \mathrm{Arc}_\eta(u, v) := \cos(\eta\Omega)\, u + \sin(\eta\Omega)\, e(u, v), & c < -1 + \epsilon_\Omega, \\ \mathrm{Slerp}_\eta(u, v), & \text{otherwise.} \end{cases} \tag{12}$$

The last branch is the exact step, and the middle one makes the unavoidable choice of arc near an antipode explicit. The first branch is what the solver executes in the convergence regime, and the following lemma is what the boundedness argument of Section 3.2.1 needs from it.

Lemma 2 (The guarded step stays on the sphere and on the arc). *Let $u, v \in \mathbb{S}^{d-1}$ and $\eta \in [0, 1]$. Every branch of eq. (12) returns a unit vector. On the first branch, $\mathrm{nlerp}_\eta(u, v)$ lies on the same great circle as $\mathrm{Slerp}_\eta(u, v)$, at angle $\theta = \eta\Omega + O(\Omega^3)$ from u, where $\Omega = \angle(u, v)$.*

Proof. The first branch normalizes explicitly, the third has unit norm by Lemma 1, and the middle branch is $\cos(\eta\Omega)u + \sin(\eta\Omega)e$ with $e \perp u$ of unit norm. For the first branch, $(1-\eta)u + \eta v$ lies in $\mathrm{span}\{u, v\}$, so normalizing it lands on the great circle through u and v: the guard can change the position along the arc, not the arc. Its angle θ from u satisfies

$$\cos\theta = \frac{(1-\eta) + \eta\cos\Omega}{\|(1-\eta)u + \eta v\|_2} = \frac{1 - \eta\Omega^2/2 + O(\Omega^4)}{1 - \eta(1-\eta)\Omega^2/2 + O(\Omega^4)} = 1 - \frac{\eta^2\Omega^2}{2} + O(\Omega^4),$$

using $\cos\Omega = 1 - \Omega^2/2 + O(\Omega^4)$ and $\|(1-\eta)u + \eta v\|_2^2 = 1 - 2\eta(1-\eta)(1-\cos\Omega)$. Hence $\theta = \eta\Omega + O(\Omega^3)$: the guard's error is a third-order timing error along the correct arc. The guard fires for $\Omega \lesssim \sqrt{2\epsilon_\Omega}$, where this discrepancy is of order $\epsilon_\Omega^{3/2}$, while the $0/0$ is removed. □

A.3 Euclidean mixing follows the same geodesic

This is the identity behind the geodesic injection of eq. (8): FPRM's affine input mixing, once normalized, moves along the same arc with a state-dependent fraction.

Lemma 3 (Normalized Euclidean mixing follows the same geodesic)**.** *Let $u, v \in \mathbb{R}^d \setminus \{0\}$, write $\widehat{u} = u/\|u\|_2$ and $\widehat{v} = v/\|v\|_2$, and assume $\widehat{v} \neq -\widehat{u}$. For any $\alpha, \beta > 0$, if $\Omega = \angle(\widehat{u}, \widehat{v}) \in (0, \pi)$, then*

$$\begin{aligned} \mathrm{Norm}(\alpha u + \beta v) &= \mathrm{Slerp}_{\lambda_{\text{eff}}}(\widehat{u}, \widehat{v}), \\ \lambda_{\text{eff}} &= \frac{1}{\Omega}\,\mathrm{atan2}(\beta\|v\|_2 \sin\Omega,\ \alpha\|u\|_2 + \beta\|v\|_2\cos\Omega) \in (0,1). \end{aligned} \tag{13}$$

If $\Omega = 0$, then $\mathrm{Norm}(\alpha u + \beta v) = \widehat{u} = \widehat{v}$.

Proof. If $\Omega = 0$, then $\widehat{u} = \widehat{v}$ and the positive mixture $\alpha u + \beta v = (\alpha\|u\|_2 + \beta\|v\|_2)\widehat{u}$ normalizes to $\widehat{u}$. This is also $\mathrm{Slerp}_s(\widehat{u}, \widehat{u})$ for every $s \in [0,1]$.

For $0 < \Omega < \pi$, define

$$e := \frac{\widehat{v} - \cos\Omega\,\widehat{u}}{\sin\Omega}.$$

As in Lemma 1, $\widehat{u}$ and e are orthonormal and $\widehat{v} = \cos\Omega\,\widehat{u} + \sin\Omega\,e$. Hence

$$\alpha u + \beta v = (\alpha\|u\|_2 + \beta\|v\|_2\cos\Omega)\,\widehat{u} + \beta\|v\|_2\sin\Omega\,e.$$

By the definition of $\mathrm{atan2}$ and λ_{eff} in the lemma, the normalized mixture is therefore

$$\mathrm{Norm}(\alpha u + \beta v) = \cos(\lambda_{\text{eff}}\Omega)\,\widehat{u} + \sin(\lambda_{\text{eff}}\Omega)\,e.$$

The first argument of $\mathrm{atan2}$ is positive, so $0 < \lambda_{\text{eff}}\Omega < \pi$. Moreover,

$$\sin(\Omega - \lambda_{\text{eff}}\Omega) = \frac{\alpha\|u\|_2\sin\Omega}{\|\alpha u + \beta v\|_2} > 0.$$

Since $\Omega - \lambda_{\text{eff}}\Omega \in (-\pi, \pi)$, this implies $0 < \lambda_{\text{eff}}\Omega < \Omega$, or $0 < \lambda_{\text{eff}} < 1$. The representation in Lemma 1 now gives

$$\mathrm{Norm}(\alpha u + \beta v) = \mathrm{Slerp}_{\lambda_{\text{eff}}}(\widehat{u}, \widehat{v})\,,$$

which is exactly eq. (13).

At $\Omega = \pi$, the shortest great-circle direction is not unique. If the two opposing weighted vectors also have equal magnitudes, their Euclidean mixture is zero and cannot be normalized. This explains why antipodal directions are excluded. □

A.4 Detailed proof of fixed-point existence

Proof of Proposition 1. Set $R := (1-\lambda)\pi$. Since $0 < \lambda < 1$, we have $0 < R < \pi$. Recall the cap

$$\mathcal{C}_{\mathbf{x}} = \{\mathbf{y} \in \mathcal{S} : \angle(y_t, x_t) \le R \text{ for every } t\}\,.$$

The exact recurrent map F_θ does not generally map this cap into itself, because the network is applied after the token-wise interpolation in eq. (8). The proof therefore uses the reversed order only in an auxiliary map:

$$H(\mathbf{y}) := \mathrm{Slerp}_\lambda(f_\theta(\mathbf{y}), \mathbf{x})\,.$$

For every $\mathbf{y} \in \mathcal{C}_{\mathbf{x}}$, the assumptions ensure that $f_\theta(\mathbf{y})_t$ is a unit vector different from $-x_t$. By Lemma 1,

$$\begin{aligned}\angle(H(\mathbf{y})_t, x_t) &= (1-\lambda)\angle(f_\theta(\mathbf{y})_t, x_t) \\ &< (1-\lambda)\pi = R.\end{aligned}$$

Thus $H(\mathcal{C}_{\mathbf{x}}) \subseteq \mathcal{C}_{\mathbf{x}}$. Continuity of f_θ and exclusion of the antipode make H continuous throughout this cap.

We next give the Euclidean coordinates needed for Brouwer's theorem. The spherical logarithm below is a vector-valued coordinate map, not a scalar logarithm. Given unit vectors u and $v \neq -u$, let $\Omega = \angle(u, v) < \pi$ and define

$$\mathrm{Log}_u(v) := \begin{cases} 0, & v = u, \\ \dfrac{\Omega}{\sin\Omega}\,(v - \cos\Omega\, u)\,, & v \neq u. \end{cases}$$

The vector in parentheses is orthogonal to u and has norm $\sin\Omega$. Therefore

$$\langle \mathrm{Log}_u(v),\, u\rangle = 0, \qquad \|\mathrm{Log}_u(v)\|_2 = \Omega.$$

In words, $\mathrm{Log}_u(v)$ points in the tangent direction of the shortest arc from u to v, and its length equals that arc's angle. Its inverse is

$$\mathrm{Exp}_u(w) := \begin{cases} u, & w = 0, \\ \cos(\|w\|_2)u + \sin(\|w\|_2)\dfrac{w}{\|w\|_2}, & w \neq 0, \end{cases}$$

for $\langle w,\, u\rangle = 0$ and $\|w\|_2 < \pi$. The two maps are continuous inverses on every closed cap of radius strictly smaller than π.

Applying the spherical logarithm token by token identifies $\mathcal{C}_{\mathbf{x}}$ continuously and bijectively with

$$\mathcal{B} := \prod_{t=1}^{T} \left\{ u_t \in \mathbb{R}^d : \langle u_t,\, x_t\rangle = 0,\ \|u_t\|_2 \le R \right\}.$$

Each factor is an ordinary closed Euclidean ball inside the hyperplane perpendicular to x_t. Consequently, $\mathcal{B}$ is nonempty, compact, and convex and can be identified with a subset of $\mathbb{R}^{T(d-1)}$.

We now apply ***Brouwer's fixed-point theorem*** (Brouwer, 1911), in the following standard form: if K is a nonempty, compact, convex subset of a finite-dimensional Euclidean space and $G : K \to K$ is continuous, then

$$\text{there exists } y^\star \in K \text{ such that } G(y^\star) = y^\star.$$

For a textbook statement, see Ortega & Rheinboldt (1970, Section 6.3, p. 161). Transforming H into the logarithmic coordinates above produces just such a continuous self-map of $\mathcal{B}$. Therefore there exists $\mathbf{y}^\star \in \mathcal{C}_{\mathbf{x}}$ such that

$$\mathbf{y}^\star = H(\mathbf{y}^\star) = \mathrm{Slerp}_\lambda(f_\theta(\mathbf{y}^\star), \mathbf{x})\,.$$

Now define

$$\mathbf{z}^\star := f_\theta(\mathbf{y}^\star).$$

The preceding fixed-point identity gives $\mathbf{y}^\star = \mathrm{Slerp}_\lambda(\mathbf{z}^\star, \mathbf{x})$, and hence

$$\begin{aligned} F_\theta(\mathbf{z}^\star; \mathbf{x}) &= f_\theta(\mathrm{Slerp}_\lambda(\mathbf{z}^\star, \mathbf{x})) \\ &= f_\theta(\mathbf{y}^\star) = \mathbf{z}^\star. \end{aligned}$$

Thus $\mathbf{z}^\star$ is a fixed point of the exact injection-first recurrent map. Notice that the proof places $\mathbf{y}^\star$, not necessarily $\mathbf{z}^\star$, in the input-centered cap. □

A.5 THE FIRST DERIVATIVE OF SLERP AT COINCIDENT ENDPOINTS

Lemma 4 (First-order interpolation rule)**.** *Fix $0 \le \eta \le 1$ and $a \in \mathbb{S}^{d-1}$. As unit vectors u, v approach a,*

$$\mathrm{Slerp}_\eta(u, v) = (1-\eta)u + \eta v + O(\|u - v\|_2^2),$$

with the exact value $\mathrm{Slerp}_\eta(u, u) = u$ at coincidence. In particular, the interpolation is jointly differentiable at (a, a), with tangent derivative

$$D\,\mathrm{Slerp}_\eta(a, a)[h, k] = (1-\eta)h + \eta k, \qquad \langle a,\, h\rangle = \langle a,\, k\rangle = 0.$$

Proof. For $\Omega = \angle(u, v) > 0$ small, the elementary estimate $\sin(c\Omega) = c\Omega + O(\Omega^3)$, for fixed c, gives

$$\frac{\sin((1-\eta)\Omega)}{\sin\Omega} = (1-\eta) + O(\Omega^2), \qquad \frac{\sin(\eta\Omega)}{\sin\Omega} = \eta + O(\Omega^2).$$

Substituting into the interpolation formula gives an error of order Ω^2, since both endpoints have unit norm. The chord identity $\|u - v\|_2 = 2\sin(\Omega/2)$ implies $\Omega = O(\|u - v\|_2)$, proving the stated estimate. If $u = v$, the error is exactly zero because $\mathrm{Slerp}_\eta(u, u) = u$.

To obtain the joint derivative, express nearby endpoints using tangent coordinates $h, k \perp a$:

$$u = \frac{a+h}{\|a+h\|_2} = a + h + O(\|h\|_2^2), \qquad v = \frac{a+k}{\|a+k\|_2} = a + k + O(\|k\|_2^2).$$

Here $\|a+h\|_2^2 = 1 + \|h\|_2^2$, so normalization has no first-order effect. The uniform interpolation estimate now yields

$$\mathrm{Slerp}_\eta(u, v) = a + (1-\eta)h + \eta k + O(\|h\|_2^2 + \|k\|_2^2).$$

Thus the linear response to arbitrary simultaneous endpoint perturbations is $(1-\eta)h + \eta k$, as claimed. □

A.6 PROOF OF LOCAL CONVERGENCE UNDER ANGULAR DAMPING

Proof of Theorem 1. **Tangent directions and fixed points.** Let $J = D_{\mathbf{z}} F_\theta(\mathbf{z}^\star; \mathbf{x})$. Consider a differentiable spherical curve $\mathbf{z}(\varepsilon)$ with $\mathbf{z}(0) = \mathbf{z}^\star$ and $\mathbf{z}'(0) = \mathbf{u}$. Both its input and output tokens have unit norm, so differentiation gives

$$0 = \frac{d}{d\varepsilon}\|z_t(\varepsilon)\|_2^2\Big|_{\varepsilon=0} = 2\langle z_t^\star,\, u_t\rangle,$$
$$0 = \frac{d}{d\varepsilon}\|F_\theta(\mathbf{z}(\varepsilon); \mathbf{x})_t\|_2^2\Big|_{\varepsilon=0} = 2\langle z_t^\star,\, (J\mathbf{u})_t\rangle.$$

The output derivative uses $F_\theta(\mathbf{z}^\star; \mathbf{x}) = \mathbf{z}^\star$. Hence J sends tangent perturbations to tangent perturbations at the same fixed point; it includes all coupling between tokens.

For $\eta > 0$, interpolation moves a positive fraction of the shortest arc, so it returns its first endpoint exactly when the endpoints coincide. The damped update therefore has exactly the fixed points of F_θ, wherever these arcs are defined. At $\mathbf{z}^\star$ the endpoints coincide, and continuity ensures that nearby endpoint pairs are non-antipodal.

Linearization and the damping range. Represent a nearby state by tangent coordinates $\mathbf{w} = (w_t)_{t=1}^T$:

$$z_t = \frac{z_t^\star + w_t}{\|z_t^\star + w_t\|_2}, \qquad \langle z_t^\star,\, w_t\rangle = 0.$$

These coordinates measure displacement along the sphere to first order: $\mathbf{z} = \mathbf{z}^\star + \mathbf{w} + O(\|\mathbf{w}\|^2)$. The inverse is $w_t = z_t/\langle z_t^\star,\, z_t\rangle - z_t^\star$, defined nearby, and also has identity tangent derivative. Differentiability of F_θ gives

$$F_\theta(\mathbf{z}; \mathbf{x}) = \mathbf{z}^\star + J\mathbf{w} + o(\|\mathbf{w}\|).$$

In particular, the two interpolation endpoints differ by $O(\|\mathbf{w}\|)$. By Lemma 4, replacing their spherical interpolation by their weighted average incurs only $O(\|\mathbf{w}\|^2)$ error. Thus, writing Φ_η for the damped update in these coordinates,

$$\Phi_\eta(\mathbf{w}) = A\mathbf{w} + o(\|\mathbf{w}\|), \qquad A := (1-\eta)\,\mathrm{Id} + \eta J.$$

If $Jv = \zeta v$ for a nonzero, possibly complex eigenvector v, then $Av = [1 + \eta(\zeta - 1)]v$. Conversely, every eigenvector of A is one of $J = (A - (1-\eta)\,\mathrm{Id})/\eta$. This identifies all eigenvalues without assuming diagonalizability. As in Euclidean damping (Movahedi et al., 2026, Appendix A.3),

$$|1 + \eta(\zeta - 1)|^2 = 1 + 2\eta(\mathrm{Re}(\zeta) - 1) + \eta^2|\zeta - 1|^2 < 1$$

exactly when $0 < \eta < 2(1 - \mathrm{Re}(\zeta))/|\zeta - 1|^2$. The theorem's damping bound therefore places every eigenvalue of A strictly inside the unit disk.

From the linearization to convergence. This eigenvalue condition need not make the Euclidean norm shrink at every step. A standard finite-dimensional stability result instead provides a vector norm $\|\cdot\|_*$ with induced matrix norm $a := \|A\|_* < 1$ (Ortega & Rheinboldt, 1970). This norm is equivalent to the Euclidean norm: positive constants c_1, c_2 satisfy

$$c_1\|\mathbf{w}\| \le \|\mathbf{w}\|_* \le c_2\|\mathbf{w}\|.$$

It therefore measures the same notion of convergence, up to fixed factors.

Choose $a < q < 1$. The remainder $o(\|\mathbf{w}\|)$ means that its norm divided by $\|\mathbf{w}\|$ tends to zero. By norm equivalence, we may choose a sufficiently small neighborhood where

$$\|\Phi_\eta(\mathbf{w}) - A\mathbf{w}\|_* \le (q - a)\|\mathbf{w}\|_*.$$

The triangle inequality then gives the full one-step error bound:

$$\begin{aligned}\|\Phi_\eta(\mathbf{w})\|_* &\le \|A\mathbf{w}\|_* + \|\Phi_\eta(\mathbf{w}) - A\mathbf{w}\|_* \\ &\le a\|\mathbf{w}\|_* + (q - a)\|\mathbf{w}\|_* = q\|\mathbf{w}\|_*.\end{aligned}$$

Because $q < 1$, a sufficiently small ball in this norm stays within itself after every update. Starting there, induction gives $\|\mathbf{w}_i\|_* \le q^i\|\mathbf{w}_0\|_*$. Finally, the coordinate map and its inverse are smooth nearby, so they bound coordinate error and state error by constant multiples of each other. Together with norm equivalence, this yields

$$\|\mathbf{z}_i - \mathbf{z}^\star\| \le Cq^i\|\mathbf{z}_0 - \mathbf{z}^\star\|$$

for some $C > 0$, proving the asserted local geometric convergence. □

The guarded step. Near $\mathbf{z}^\star$ the endpoints $\mathbf{z}_i$ and $F_\theta(\mathbf{z}_i; \mathbf{x})$ are $O(\|\mathbf{w}\|)$ apart, so on a neighborhood of $\mathbf{z}^\star$ the solver executes the first branch of eq. (12). By Lemma 2 that branch lies on the same arc as the exact step, at an angle from u that differs by $O(\Omega^3)$, so it differs from $\mathrm{Slerp}_\eta(u, v)$ by $O(\|u - v\|_2^3)$ and shares the expansion $(1-\eta)u + \eta v + O(\|u - v\|_2^2)$ of Lemma 4. It also returns u only when $v = u$, because for $\eta > 0$ a normalized nonnegative combination of two non-antipodal unit vectors equals u only if $v = u$. The proof above therefore applies verbatim, and Theorem 1 holds for the guarded step the solver runs.

Remark 1 (Optimal damping, and when the window is narrow). Write $\zeta = a + bi$. The squared modulus $|1+\eta(\zeta-1)|^2 = 1-2\eta(1-a)+\eta^2|\zeta-1|^2$ is an upward parabola in η with value 1 at $\eta = 0$, so for $a < 1$ the window $0 < \eta < 2(1-a)/|\zeta-1|^2$ of eq. (11) is exactly the interval on which it lies below one, the vertex $\eta^\star = (1-a)/|\zeta-1|^2$ is the optimal damping for that mode, and the smallest attainable squared modulus is $1-(1-a)^2/|\zeta-1|^2$, a contraction factor of $\sqrt{1-(1-a)^2/|\zeta-1|^2}$ per step. Three consequences. *(i)* Damping handles rotation but not expansion: for $a \ge 1$ every term is nonnegative, so no $\eta > 0$ contracts the mode and only training can remove it. *(ii)* The guarantee is weak when $1 - a$ is small compared with $|b|$, that is, when $\zeta - 1$ is nearly imaginary, whatever $|\zeta|$ is: for $\zeta = 0.9 + 1.5i$ the undamped mode grows by $|\zeta| \approx 1.75$ per step while rotating by about $59°$, the window is $0 < \eta < 0.089$, and even the optimal damping contracts by only 0.22% per step, so halving the error takes about 313 iterations. Bounding the rotation magnitude, not only the expansion, is therefore what makes halting fast. *(iii)* The patience rule of Algorithm 1, which starts at $\eta = 1$ and multiplies it by γ whenever the residual stalls, is a derivative-free descent into

this window: well-conditioned modes ($\zeta = 0.5$ admits every $\eta < 4$) converge at once, and the geometric decay eventually drops η below the threshold of the rotational modes, at which point their tangent error contracts geometrically and, whenever the residual then improves within P iterations, the decay pauses, without any spectrum ever being estimated. Because the contraction is in an adapted norm, the residual need not fall monotonically, so the rule may keep shrinking η; this slows convergence but cannot break it, since every smaller η stays inside the window.

A.7 Global convergence under sufficiently strong input injection

Proof of Theorem 2. We use the token-wise logarithm and exponential from Appendix A.4. They satisfy

$$\mathrm{Slerp}_\lambda(\mathbf{z}, \mathbf{x}) = \mathrm{Exp}_{\mathbf{x}}((1-\lambda)\,\mathrm{Log}_{\mathbf{x}}(\mathbf{z}))$$

whenever $z_t \neq -x_t$ for every token t. Indeed, eq. (6) gives $\mathrm{Slerp}_\lambda(z_t, x_t) = \mathrm{Slerp}_{1-\lambda}(x_t, z_t)$, which by the representation in Lemma 1 equals $\cos((1-\lambda)\Omega)\,x_t + \sin((1-\lambda)\Omega)\,e$ with $\Omega = \angle(x_t, z_t)$ and $\mathrm{Log}_{x_t}(z_t) = \Omega e$, and this is $\mathrm{Exp}_{x_t}((1-\lambda)\,\mathrm{Log}_{x_t}(z_t))$ by definition, also when $\Omega = 0$. Thus input injection multiplies the logarithmic coordinates by exactly $1-\lambda$: it preserves the direction toward the current state while reducing its angular distance from the input.

The geometric difficulty is that shrinking angles to the input does not give a global contraction bound between states. Two states approaching an input antipode along different arcs can be arbitrarily close while their injected states remain separated for fixed $0 < \lambda < 1$. Thus the direct global Lipschitz argument for Euclidean input mixing in FPRM (Movahedi et al., 2026, Appendix A.2) does not apply here. Instead, every injected token has angle less than $(1-\lambda)\pi$ to its input. We use this uniform bound to place all injected states in one fixed neighborhood, prove contraction in its logarithmic coordinates, and then transfer convergence back to the original states.

These are smooth inverse coordinate maps for $\|w_t\|_2 < \pi$. The spherical exponential is smooth, and its differential has singular values 1 in the radial direction and $\sin r/r$ in the transverse directions at radius $0 < r < \pi$, with identity differential at the origin. Its differential is therefore invertible throughout this domain, so the inverse function theorem gives smoothness of the logarithm.

A fixed neighborhood of the input. Since f_θ is C^1 near $\mathbf{x}$ and $f_\theta(\mathbf{x})_t \neq -x_t$ for every t, continuity allows us to choose $0 < r < \pi/2$ such that $f_\theta \circ \mathrm{Exp}_{\mathbf{x}}$ is C^1 and none of its output tokens equals $-x_t$ on an open neighborhood of

$$\mathcal{B}_r := \left\{\mathbf{w} \in (\mathbb{R}^d)^T : \langle w_t,\, x_t\rangle = 0, \quad \|w_t\|_2 \le r \text{ for every } t\right\}.$$

Here $\mathcal{B}_r$ is a compact, convex product of ordinary tangent-space balls. Define, on this neighborhood,

$$A(\mathbf{w}) := \mathrm{Log}_{\mathbf{x}}(f_\theta(\mathrm{Exp}_{\mathbf{x}}(\mathbf{w}))).$$

Here $\mathbf{w}$ represents an injected state, and $A(\mathbf{w})$ represents the network's output in the same coordinates. Multiplication by $1-\lambda$ then performs the next input injection. Both A and $f_\theta \circ \mathrm{Exp}_{\mathbf{x}}$ are C^1, so their derivatives are bounded on the compact set $\mathcal{B}_r$. Integrating these derivatives along the line segment between any $\mathbf{w}, \mathbf{v} \in \mathcal{B}_r$ gives a common finite constant $L \ge 1$ such that

$$\|A(\mathbf{w}) - A(\mathbf{v})\| \le L\|\mathbf{w} - \mathbf{v}\|,$$
$$\|f_\theta(\mathrm{Exp}_{\mathbf{x}}(\mathbf{w})) - f_\theta(\mathrm{Exp}_{\mathbf{x}}(\mathbf{v}))\| \le L\|\mathbf{w} - \mathbf{v}\|.$$

All norms here are the product Euclidean norm defined in the main text. Moreover, $\|A(\mathbf{w})_t\|_2 < \pi$ for every token, because the logarithm's length is the angle to x_t and the antipode is excluded.

Strong injection gives a contraction. Choose

$$\lambda_0 := 1 - \min\left\{\frac{r}{\pi}, \frac{1}{2L}\right\} \in (0, 1).$$

Fix any $\lambda \in (\lambda_0, 1)$, and set $a := 1-\lambda$ and $q := aL$. Then $a\pi < r$ and $0 < q < 1/2$. The two terms in the threshold serve separate purposes: $a\pi < r$ keeps all injected states in the chosen neighborhood, while $aL < 1$ makes their distances contract. The factor $1/2$ simply leaves a strict margin below one. The map $\mathbf{w} \mapsto aA(\mathbf{w})$ sends $\mathcal{B}_r$ into itself, since

$$\|aA(\mathbf{w})_t\|_2 < a\pi < r.$$

It is also a contraction:

$$\|aA(\mathbf{w}) - aA(\mathbf{v})\| \le q\|\mathbf{w} - \mathbf{v}\|.$$

The set $\mathcal{B}_r$ is closed in a finite-dimensional Euclidean space, hence complete. Banach's contraction theorem therefore gives a unique $\mathbf{w}^\star \in \mathcal{B}_r$ satisfying $\mathbf{w}^\star = aA(\mathbf{w}^\star)$, and every iteration $\mathbf{w}_{i+1} = aA(\mathbf{w}_i)$ starting in $\mathcal{B}_r$ satisfies

$$\|\mathbf{w}_i - \mathbf{w}^\star\| \le q^i\|\mathbf{w}_0 - \mathbf{w}^\star\|.$$

Return to the original recurrence. Let $\mathbf{z}_0$ be any admissible initialization, with $z_{0,t} \neq -x_t$ for every t, and set

$$\mathbf{w}_0 := a\,\mathrm{Log}_{\mathbf{x}}(\mathbf{z}_0).$$

Because $\|w_{0,t}\|_2 < a\pi < r$, this initial coordinate lies in $\mathcal{B}_r$, regardless of the distance from $\mathbf{z}_0$ to a fixed point. The original recurrence and the coordinate iteration are related by

$$\mathbf{z}_{i+1} = f_\theta(\mathrm{Exp}_{\mathbf{x}}(\mathbf{w}_i)), \qquad \mathbf{w}_{i+1} = a\,\mathrm{Log}_{\mathbf{x}}(\mathbf{z}_{i+1}) = aA(\mathbf{w}_i).$$

These identities hold inductively for every $i \ge 0$: each $\mathbf{w}_i$ remains in $\mathcal{B}_r$, where the network's outputs avoid the antipodes. Hence every subsequent interpolation is well-defined.

Define $\mathbf{z}^\star := f_\theta(\mathrm{Exp}_{\mathbf{x}}(\mathbf{w}^\star))$. The identity $\mathbf{w}^\star = aA(\mathbf{w}^\star)$ implies $\mathbf{w}^\star = a\,\mathrm{Log}_{\mathbf{x}}(\mathbf{z}^\star)$, so

$$F_\theta(\mathbf{z}^\star; \mathbf{x}) = f_\theta(\mathrm{Exp}_{\mathbf{x}}(\mathbf{w}^\star)) = \mathbf{z}^\star.$$

This fixed point is admissible by the choice of r. Conversely, any admissible fixed point $\mathbf{z}$ of F_θ gives $\mathbf{w} = a\,\mathrm{Log}_{\mathbf{x}}(\mathbf{z}) \in \mathcal{B}_r$ with $\mathbf{w} = aA(\mathbf{w})$. Uniqueness of $\mathbf{w}^\star$ then implies $\mathbf{z} = \mathbf{z}^\star$.

Finally, the Lipschitz bound above transfers convergence to the original states:

$$\begin{aligned}\|\mathbf{z}_{i+1} - \mathbf{z}^\star\| &\le L\|\mathbf{w}_i - \mathbf{w}^\star\| \\ &\le Lq^i\|\mathbf{w}_0 - \mathbf{w}^\star\| \le 2Lr\sqrt{T}\,q^i, \qquad i \ge 0.\end{aligned}$$

Thus the theorem holds with $C = 2Lr\sqrt{T}$, uniformly over all admissible initializations. The neighborhood, derivative bound, and injection threshold depend on the fixed network and input. □

B Details on Experimental Setup

This appendix section provides the details on the experimental setup and experiments in Section 4. Details on tasks and data are in Appendix B.1, on models in Appendix B.2, on training and solver settings in Appendix B.3. Table 3 reports all hyperparameters. The evaluation protocol is in Appendix B.4.

B.1 Tasks and data

Sudoku-Extreme. This dataset consists of 9×9 Sudoku puzzles selected for difficulty, with a test split of $422{,}786$ puzzles. The standard training build has $1{,}000$ base puzzles, each augmented $1000\times$ by validity-preserving transformations (relabeling of the digits and permutations of rows, columns, bands, and stacks), giving 10^6 training examples. We train instead on a $5\text{k}\times200$ build, $5{,}000$ base puzzles with 200 augmentations each, which has the same number of training examples. Unless stated otherwise, every SFPRM Sudoku run and every rollout of Sections 4.2 and 4.3 uses this build. A puzzle is a sequence of 81 cell tokens (blank or a digit) preceded by a learned per-puzzle embedding of 16 tokens, and the model predicts the digit of every cell.

Maze-Hard. The 30×30 maze benchmark of the same suite: $1{,}000$ training and $1{,}000$ test mazes, used without augmentation. A maze is a sequence of 900 cell tokens (wall, free, start, goal) plus the puzzle embedding, and the model predicts for every cell whether it lies on the shortest path.

ARC-AGI-1 and ARC-AGI-2. The abstraction-and-reasoning benchmarks of Chollet et al. (2024): few-shot tasks in which a grid transformation must be inferred from demonstration pairs and applied to a test grid. Following the HRM/TRM protocol, we train on 960 (ARC-1) and $1{,}280$ (ARC-2) base puzzles, each augmented $1000\times$ with the transformations of Wang et al. (2025), and report Pass@2 on the 400 and 120 evaluation tasks. Grids are padded to 30×30, so a puzzle is a sequence of 900 cell tokens over a 12-token vocabulary plus the puzzle embedding.

B.2 MODELS

SFPRM is implemented in the FPRM code base and share an identical backbone with 6.83M trainable parameters: hidden size 512, 8 attention heads, MLP expansion 4, rotary position embeddings, and the configuration fields `H_cycles`/`L_cycles`/`L_layers` $= 3/6/2$ of the code base, run as a single weight-tied loop without hierarchy. The number of iterations backpropagated through, is the BPTT window $K = 6$ of Section 3.1.

FPRM. FPRM uses a pre-norm block with the coupled residual-rescaling coefficients $(\alpha_1, \beta_1, \alpha_2, \beta_2)$ of Section 3.1, linear damping, and the relative residual r_{E} with tolerance $\tau_{\mathrm{E}} = 0.1$.

SFPRM. The spherical model replaces the block by the nGPT-style block of Section 3.2: QK-normalized attention and a SwiGLU MLP, each folded into the unit-norm state by an eigen-learning-rate retraction, with all weight vectors renormalized after every optimizer step. Three further choices are specific to SFPRM. *(i)* The input is injected by the geodesic step of eq. (8) with a learned fraction λ, initialized at 0.1. *(ii)* The readout is *unnormalized*: the rows of the output head are not constrained to unit norm, since with a unit-norm state that constraint would cap every logit at $\pm s_z$, and with it the confidence of the predictions. *(iii)* The solver is the geodesic one of Algorithm 1, with the chordal residual of eq. (9) and tolerance $\tau_c = 0.01$, which corresponds to an angular tolerance of $\tau_{\mathrm{ang}} = \arccos(0.99) \approx 8.1^\circ$ per token.

B.3 TRAINING AND SOLVER SETTINGS

Table 3 lists every hyperparameter. Unless stated otherwise, SFPRM trains with a constant learning rate after the warmup steps and an exponential moving average of the weights with rate 0.999. We optimize SFPRM's 2-D weight matrices with Muon (Jordan et al., 2024; Pethick et al., 2025) at learning rate 5×10^{-3} and momentum 0.95, and its embeddings, normalization scales, and scalar parameters with AdamW at 10^{-3}. Weight decay is 0.1 for SFPRM, and the per-puzzle embeddings have their own learning rate of 10^{-3}. SFPRM additionally uses weight dropout and variational dropout of 0.1 on the main build (variational dropout only on the 1k×1000 Sudoku run). Following the HRM/FPRM code base, one *epoch* draws one augmented example for every base puzzle; hence 12,000 epochs on the 5k×200 build and 60,000 epochs on the 1k×1000 build and on Maze-Hard all amount to the same 78,125 optimizer steps at a global batch of 768. For the final Sudoku results, we train for 15,000 epochs.

Deep supervision. Training runs one segment of $K = 6$ solver iterations per optimizer step and backpropagates through all K geodesic steps of the segment; the recurrent state is carried across optimizer steps and detached between segments, and a sample is re-initialized once it halts or once it has been unrolled for `max_iter` $= 12$ iterations. The halting residual, patience counter, step-size schedule, and best-iterate bookkeeping are evaluated on detached states, so they steer the solver without entering the gradient.

Solver. Both geodesic steps of SFPRM use the guarded interpolation gSlerp of eq. (12). Per sample the solver keeps the current step size η, the patience counter, the best residual, and the best-residual state. As in FPRM, η starts at 1 and is multiplied by $\gamma = 0.99$ whenever the residual has not improved for $P = 5$ consecutive iterations. The loop halts when the residual drops below tolerance, when η falls below the floor $\eta_{\min}$, or when the iteration budget is exhausted.

B.4 EVALUATION PROTOCOL

All tasks are scored by exact accuracy (Pass@1 for Maze-Hard and Sudoku-Extreme and Pass@2 for ARC-1 and ARC-2): a puzzle counts only if every cell is correct, i.e. the whole board or the whole path. We report the final exact test accuracy, except that the SFPRM run on the 1k×1000 Sudoku build is reported at both its best and its final evaluation because it collapses between them (Table 4). The published rows of Table 2 are taken from the respective papers.

C ADDITIONAL RESULTS, ABLATIONS, AND ANALYSIS

This section provides the results that complement Section 4: the Sudoku-Extreme training dataset ablation (Appendix C.1), the Sudoku-Extreme and Maze-Hard counterparts of the convergence anal-

Table 3: **Hyperparameters and learned scalars of the SFPRM runs in Section 4**. Sudoku-Extreme values are for the epoch-15,000 run, while values in parentheses are for the 12,000-epoch constant-schedule run. All convolutions are depthwise and bias-free. For the per-channel step sizes α_A, α_M and the per-dimension QK scale s_{qk}, cells give the mean for layer 1 / layer 2, and the range row gives the minimum and maximum over all channels of both layers.

	Maze-Hard	Sudoku-Extreme	ARC-AGI-1	ARC-AGI-2
Data and schedule				
base puzzles × augmentations	1,000 × 1	5,000 × 200	960 × 1,000	1,280 × 1,000
sequence length / vocabulary	900 / 6	81 / 11	900 / 12	900 / 12
test set	1,000 mazes	422,786 puzzles	400 tasks	120 tasks
epochs	60,000	15,000 (12,000)	100,000	100,000
optimizer steps	78,125	97,656 (78,125)	518,226	722,238
global batch size	768			
Optimization				
optimizer	Muon + AdamW			
Muon lr / momentum	5×10^{-3} / 0.95			
AdamW lr	1×10^{-3}			
LR schedule	constant	cosine to 0.87× (const.)	constant	constant
warmup steps	200	200 (2,000)	0	0
weight decay	0.1			
puzzle-embedding lr / wd	10^{-2} / 0	10^{-3} / 0	10^{-2} / 0	10^{-2} / 0
EMA rate	0.999			
Backbone				
hidden / heads / MLP expansion	512 / 8 / 4			
`H_cycles` / `L_cycles` / `L_layers`	3 / 6 / 2			
BPTT window K (`n_back`)	6			
puzzle-embedding length	16			
positional encoding	RoPE			
conv branch	1-D causal, $k = 4$	none (2-D, $k = 3$)	1-D causal, $k = 4$	1-D causal, $k = 4$
trainable parameters	6.83M			
Solver and regularization				
`max_iter` (train / eval)	24 / 1,000	12 / 35,000	8 / 48	8 / 48
halting tolerance	$\tau_c = 0.01$			
step-size decay γ (train / eval)	0.9 / 0.99	0.9 / 0.99	0.9 / 0.9	0.9 / 0.9
patience P / floor $\eta_{\min}$	5 / 0.001			
readout	unnormalized head, scale s_z			
Q-head loss weight	0	0	0.5	0.5
weight dropout	0.1	0.1	0.05	0.05
variational dropout	0.1			
Learned scalars at the evaluated checkpoint				
input injection λ, init. 0.1	0.575	0.1, fixed (0.470)	0.334	0.232
logit scale s_z, init. 16	75.8	88.9 (75.8)	206.0	206.2
attention step size α_A, init. 0.5	0.50 / 0.54	0.41 / 0.50 (0.49 / 0.42)	0.44 / 0.46	0.46 / 0.45
MLP step size α_M, init. 0.5	0.48 / 0.58	0.47 / 0.47 (0.48 / 0.64)	0.49 / 0.67	0.50 / 0.76
QK scale s_{qk}, init. 1	1.56 / 1.40	1.11 / 0.95 (0.86 / 1.00)	1.31 / 1.53	1.40 / 1.49

yses of Section 4.2 (Appendix C.2), the learned scalars over training (Appendix C.3), additional results on the geometry of the computation (Appendix C.4), and the study of alternative fixed-point solvers at inference (Appendix C.5).

C.1 Training-build ablation

Table 4 compares SFPRM with an FPRM control trained under the same budget on the two Sudoku-Extreme builds described in Appendix B.1. The 5k×200 does not seem to provide benefits for FPRM, however for SFPRM it removes the collapse happening when training with the 1k×1000 build between its best and its final checkpoint.

C.2 Convergence and halting

For Sections 4.2 and 4.3 each trained model is rolled with halting disabled and without segment resets, in fp32, for a fixed budget of $N_{\max} = 400$ iterations on $n = 512$ uniformly sampled test sequences from Sudoku-Extreme and $N_{\max} = 1000$ on the full Maze-Hard test data. At every iterate

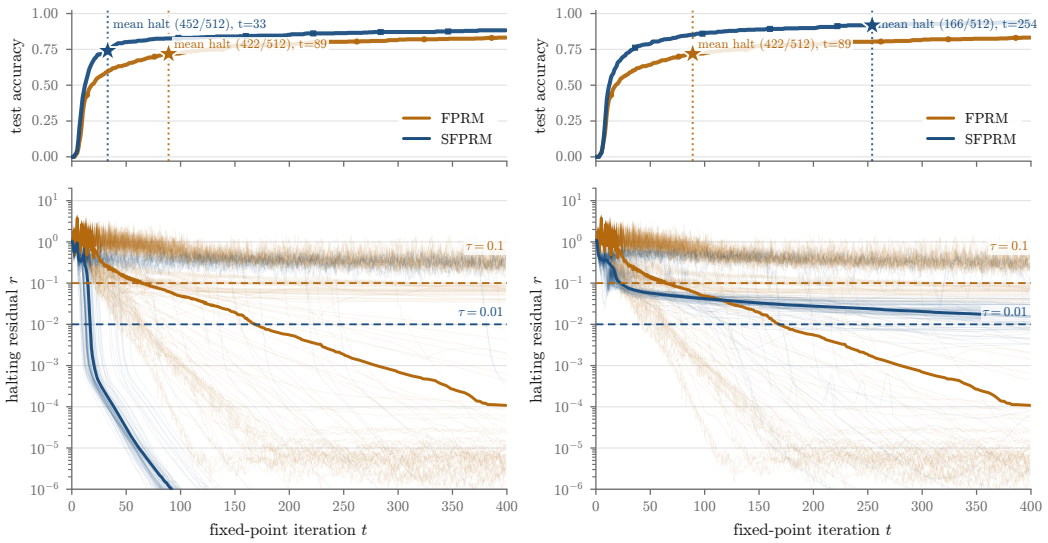


Figure 9: **Exact accuracy and halting on Sudoku-Extreme.** Open-loop rollouts on 512 test sequences of the released FPRM checkpoint (amber) and of SFPRM (blue) with the learned $\lambda = 0.47$ (left) and with λ fixed at 0.1 (right, the run in Table 2). *Top:* exact accuracy of the decoded state. Dotted lines and stars mark each model's mean halt iteration, labelled with how many sequences halt. *Bottom:* every sequence's halting residual (thin) and its median (heavy) against its own threshold (τ_c=0.01 chordal for SFPRM, τ_{E}=0.1 for FPRM).

$\mathbf{z}_i$ the model's readout head decodes the board, and we record exact and per-token accuracy together with the residual each solver tests: the worst-token chordal residual $r_{c,i} = \max_t\big(1 - \langle z_{i,t}, \widetilde{z}_{i,t}\rangle\big)$ with $\tau_c = 0.01$ for SFPRM, and the worst-token relative residual $r_{\mathrm{E},i} = \max_t \|z_{i,t} - \widetilde{z}_{i,t}\|_\infty / \|\widetilde{z}_{i,t}\|_\infty$ with $\tau_{\mathrm{E}} = 0.1$ for FPRM, where $\widetilde{\mathbf{z}}_i = F_\theta(\mathbf{z}_i; \mathbf{x})$ is each model's proposal. A sequence counts as halted at the first iteration from which its residual stays below its own threshold. Halt iterations are reported per sequence (mean over the sequences that halt), whereas the deployed evaluator halts a whole batch on its worst sequence, so the per-sequence counts are idealized costs. Every FPRM's rollout is generated using the released checkpoints from Movahedi et al. (2026).

Halting. Figure 9 shows the two Sudoku runs of Table 3 (one that reached 92.8% and one 96.6% exact accuracy) against FPRM. With the learned $\lambda = 0.47$ (left), the median residual of SFPRM drops below τ_c within a few dozen iterations, whereas FPRM's decays roughly geometrically over the whole window. 452 of the 512 sequences halt for SFPRM, at a mean of 33 iterations, against 422 at a mean of 89 for FPRM. Exact accuracy is still rising after the mean halt for both models, so on this task the threshold fires before the population has finished improving. With λ fixed at 0.1 (right) the residual of SFPRM decays far more slowly and hovers near τ_c, so only 166 sequences halt within 400 iterations, at a mean of 254, although the decoded accuracy is higher. In both runs the puzzles a solver halts on are solved in 99.7–100% of cases, so on Sudoku a halt certifies the answer, whereas a missing halt does not certify a failure.

Table 4: Exact accuracy on Sudoku-Extreme for SFPRM vs. FPRM at matched backbone (6.83M trainable parameters), data, global batch (768), and optimizer steps/epochs (78,125/12,000). "1k × 1000" and "5k×200" are the number of distinct base puzzles times the augmentations per puzzle. The SFPRM 1k run is given at both its best and checkpoint because it collapses between them, which the 5k build fixes. For the 5k run the final checkpoint is also the best. The FPRM rows fall well below the 94.2% reported for FPRM in Table 2.

Model	Setting	Sudoku-Ext.
FPRM	Sudoku 1k × 1000	78.5%
FPRM	Sudoku 5k × 200	78.4%
SFPRM	Sudoku 1k × 1000 (best checkpoint)	87.0%
SFPRM	Sudoku 1k × 1000 (final checkpoint)	12.2%
SFPRM	Sudoku 5k × 200	**92.8%**

Fixed-point uniqueness. Figure 10 repeats the global-convergence test of Figure 4 on the Maze model (learned $\lambda = 0.575$), running the raw map without damping for 1000 steps. The contractive regime of Theorem 2 exists here too, but it is narrower: all eight initializations reach the same fixed

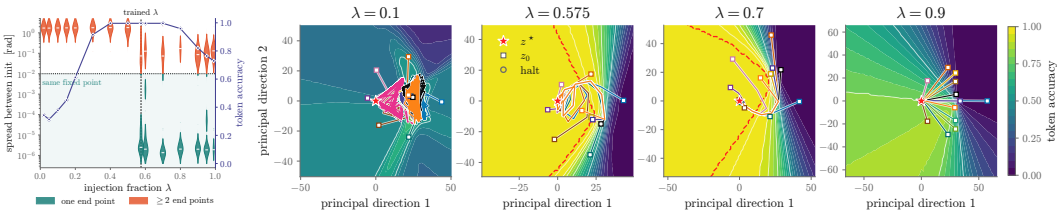
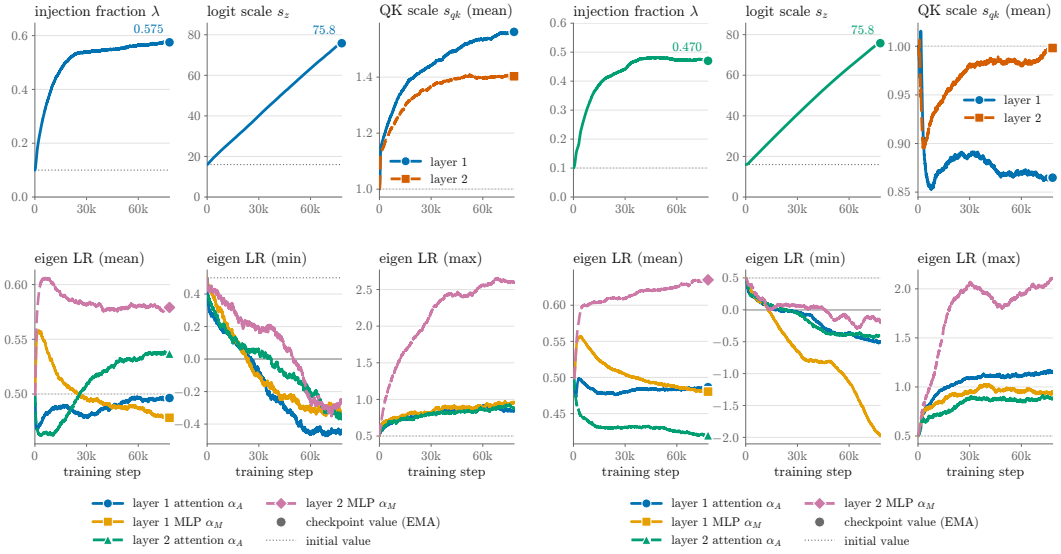


Figure 10: **Global convergence under input injection on Maze-Hard** (SFPRM, learned $\lambda = 0.575$), analogue to Figure 4. *(left)* The largest angle between the end states of eight initializations of the raw map (teal: all coincide, coral: ≥ 2 end states), and the token accuracy of the end state (indigo, right axis), as a function of λ. *(right)* Trajectories at four values of λ for a single Maze example: the top-2 principal plane of the trajectories from the eight initializations $\mathbf{z}_0$ (squares) through $\mathbf{z}^\star$ (star; circles: first iteration with $r_{c,i} < \tau_c$). The color shows the token accuracy of the state decoded at each position (dashed red: exact-match region).



(a) Maze-Hard (b) Sudoku-Extreme

Figure 11: **Learned parameters of SFPRM during training.** Per task, top row: injection fraction λ, logit scale s_z, and QK scale s_{qk} (mean over its 64 dimensions, per layer). Bottom row: the eigen learning rates α_A (attention) and α_M (MLP) of both layers, as the mean, minimum and maximum over the 512 hidden dimensions.

point only for λ between about 0.6 and 0.8. The trained λ sits at the lower edge of this band, where some mazes have a single end point and others do not, and below $\lambda \approx 0.5$ no maze does. For $\lambda \geq 0.9$ the end states spread again, while the token accuracy of the fixed point falls to 0.7–0.8 as the input pulls the state onto itself. On the maze of panel (b) the raw map does not reach the tolerance at the trained λ within 1000 steps, although it decodes the maze exactly throughout; it is the damped solver that halts this model. At $\lambda = 0.7$ the raw map converges inside the exact-match region, and at $\lambda = 0.9$ the eight paths run straight to one point that no longer solves the maze.

C.3 Eigen LR and other learnable parameters during training

Figure 11 tracks the learned scalars of SFPRM during training on Maze and Sudoku. Both models raise the injection fraction from 0.1 to $\lambda = 0.575$ on Maze and $\lambda = 0.470$ on Sudoku, with most of the increase in the first 30k of 78k training steps. This moves them toward the strong-injection regime of Theorem 2. on Sudoku, all eight initializations already reach the same fixed point on 72% of puzzles at the learned λ (Figure 4). The logit scale s_z grows almost linearly from 16 to 76, at nearly the same rate on both tasks, and is still rising when training ends. The eigen learning

rates keep means near their initial 0.5, similarly as noted by Loshchilov et al. (2025). However, individual dimensions spread widely, from negative values, which move the state away from the sub-layer output, to values up to 2.6.

C.4 The geometry of the computation

Qualitative results (Figures 6, 20 and 21). We use four test puzzles from each task for the open-loop unrolling experiment. We do this for 40 iterations and render at iterations 0, 2, 11, and 39. The decoded board is the per-cell argmax of the readout. The anchor map shows, per cell, the angle $\angle(z_{i,t}, x_t)$ between the iterate and its unit-normalized input embedding x_t (token plus position), on a colour scale normalized to the minimum and maximum angle over that puzzle's whole rollout, so that a colour means the same angle in every frame of a strip. The same quantity is computed for FPRM from its unnormalized state.

Anchor angles by cell class (Figures 7, 13 and 14). For four Sudoku and Maze test puzzles, we track the anchor angle of every cell throughout the unrolling and grouped by cell class (clues and blanks on Sudoku; walls, free cells that stay empty, free cells that become path, start, and goal on Maze). In the figures, the heavy lines are class means, whilst the thin lines the individual cell's anchor angles. The histograms on the right-hand side of each figure show the distribution at the end of the unrolling ($i = 399$ on Sudoku, 999 on Maze). For FPRM the angle is that of the normalized state.

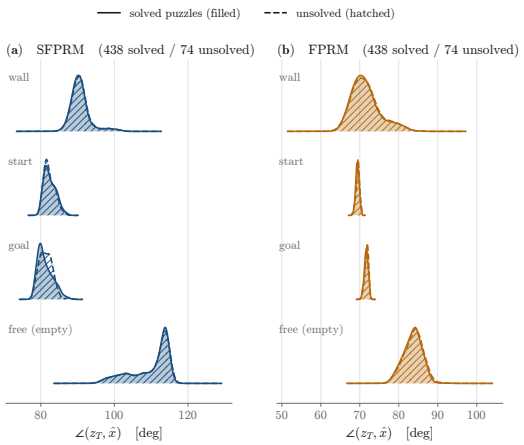


Figure 12: **Final anchor angle on solved and unsolved mazes.** The Maze-Hard analogue of Figure 8. The distribution over 512 test mazes (438 solved and 74 unsolved for both models) of each cell class's angle to its anchor at the end of the rollout, for (a) SFPRM and for (b) FPRM's normalized state. Filled: solved; Hatched: unsolved. Unlike on Sudoku, the two distributions overlap for every class, so on Maze the final angle does not predict correctness.

Trajectories on the sphere (Figure 15). We draw each token's path in the frame spanned by the inference unrolling trajectory's two leading singular directions and its input anchor $\mathbf{x}$. This way the anchor is exact rather than a near-zero projection. We only show the cell trajectories who capture the most trajectory variance. Over 98% of each trajectory's variance lies in that frame, so every token effectively moves on one great circle, as the geodesic steps of Algorithm 1 prescribe, and the anchor sits nearly orthogonal to that circle ($|\langle x_t, v_{1,2}\rangle| \approx 0.05$). The input injection pulls from far off the plane in which the computation happens, explaining how a small λ can anchor the hidden state without dictating it.

Solved versus unsolved (Figure 8). For 128 Sudoku test puzzles we compare the distribution of the final anchor angle (SFPRM) or final norm (FPRM) of clue and blank cells between solved and unsolved puzzles. The same comparison on 512 test mazes (Figure 12) shows no separation for either model.

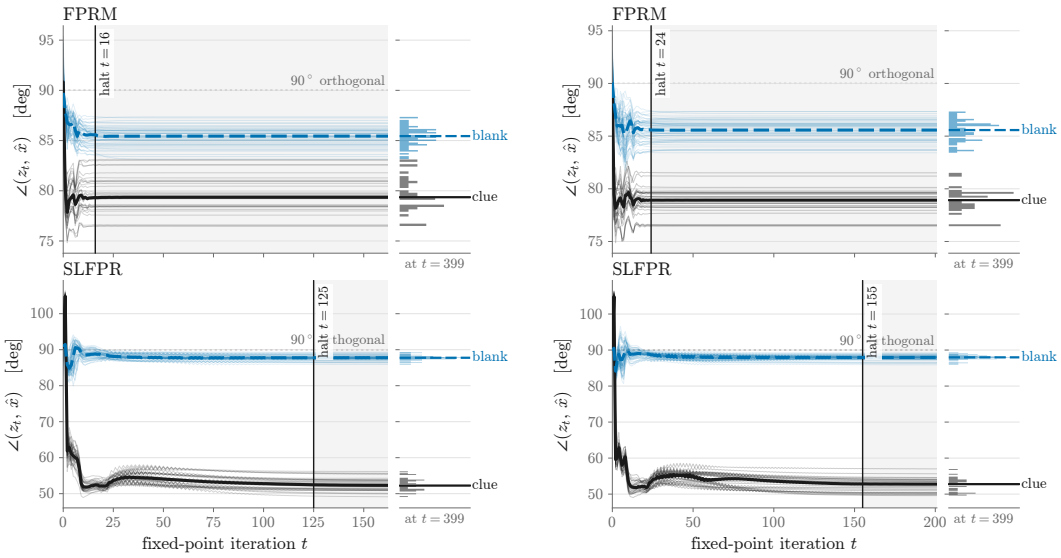
C.5 Fixed-point solvers at inference

Both SFPRM and FPRM are trained with a damped fixed-point iteration, but at inference any method that finds a fixed point of $F_\theta(\cdot\,; \mathbf{x})$ can take its place. We run four solvers (including the default one) on the trained models, without retraining and without changing the architecture (see Section 4.2).

C.5.1 Solvers

All four solvers seek a fixed point of the same trained map. The model weights are unchanged, and only the default solver (adaptive) was used during training. At test time the input embedding $\mathbf{x}$ is computed once and every solver runs the loop

$$\widetilde{\mathbf{z}}_i = F_\theta(\mathbf{z}_i; \mathbf{x}), \qquad \text{stop if } r_i < \tau, \qquad \mathbf{z}_{i+1} = \text{update}\big(\mathbf{z}_i, \widetilde{\mathbf{z}}_i, \text{history}\big),$$



(a) Sudoku test puzzle 0

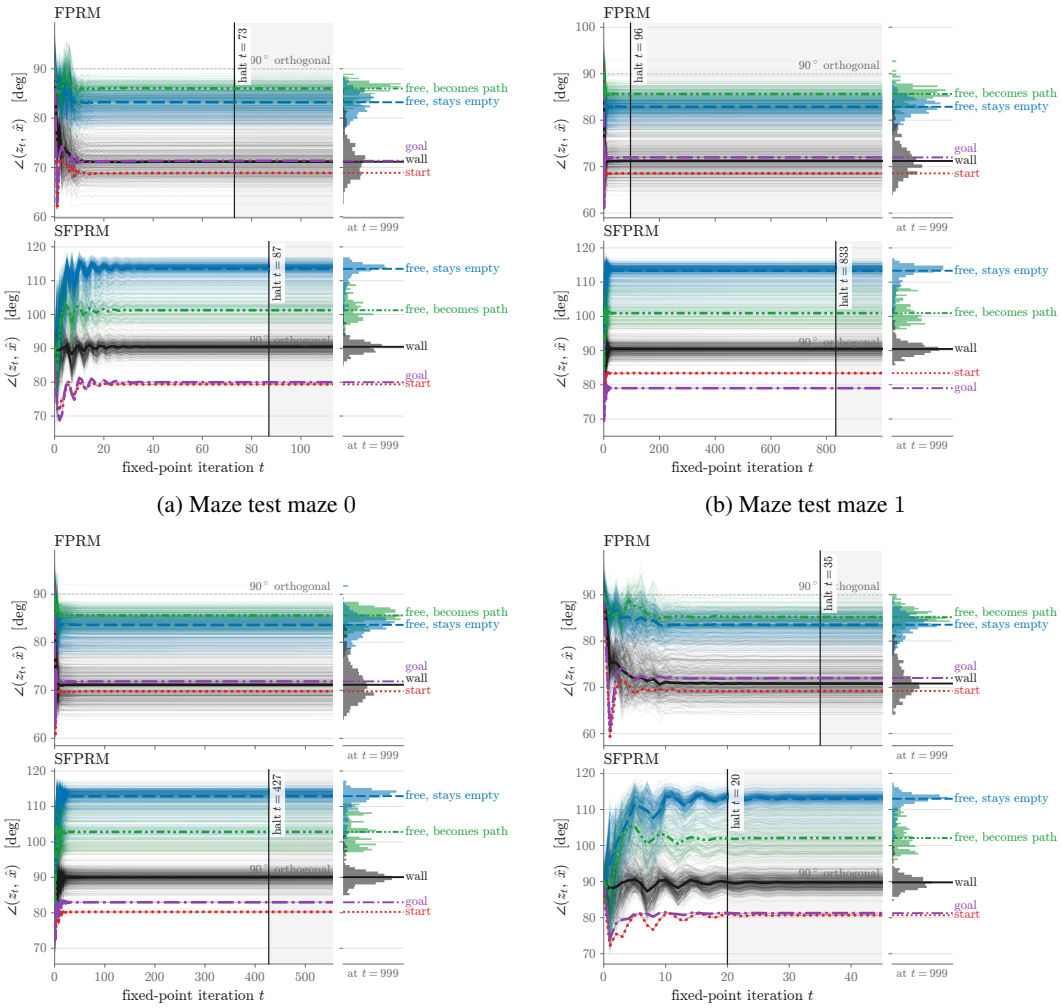
(b) Sudoku test puzzle 1

(c) Sudoku test puzzle 2

(d) Sudoku test puzzle 3

Figure 13: **Anchor angles by cell class on Sudoku-Extreme**. Four test puzzles (Maze in Figures 7 and 14). In each pair FPRM is above and SFPRM below. Thin lines are individual cells, thick lines class means, the vertical line the halt iteration, and the margin the distribution at the end of the rollout. SFPRM rotates the clues to $\approx 52°$ from their anchors within about 20 iterations and leaves the blanks near orthogonal, whereas FPRM's normalized states settle within the first few iterations with clues and blanks only $\approx 6°$ apart.

starting from a random $\mathbf{z}_0$. Across solvers, only the update function differs. Each step evaluates F_θ once, and the same evaluation gives the halting residual, so the number of steps is the compute spent. SFPRM halts on the chordal residual $r_i = r_{\mathrm{c},i}$ of eq. (9) with $\tau_c = 0.01$, and FPRM on the per-token relative residual $r_i = r_{\mathrm{E},i} = \max_t \|z_{i,t} - \widetilde{z}_{i,t}\|_\infty / \|\widetilde{z}_{i,t}\|_\infty$ (eq. (4)) with $\tau_{\mathrm{E}} = 0.1$. We write $\mathbf{g}_i := \widetilde{\mathbf{z}}_i - \mathbf{z}_i$ for the residual vector. Two operators adapt the updates to each model's state space: the interpolation $\mathcal{I}_\eta(\mathbf{a}, \mathbf{b})$ moves $\mathbf{a}$ a fraction η of the way to $\mathbf{b}$, along the great circle token by token for SFPRM, $\mathcal{I}_\eta = \mathrm{Slerp}_\eta$ as in eq. (7), and along the straight line for FPRM, $\mathcal{I}_\eta(\mathbf{a}, \mathbf{b}) = (1-\eta)\mathbf{a} + \eta\mathbf{b}$ as in eq. (3); the projection Π applies Norm to every token for SFPRM and is the identity for FPRM.



(a) Maze test maze 0

(b) Maze test maze 1

(c) Maze test maze 2

(d) Maze test maze 3 (fig. 7)

Figure 14: **Anchor angles by cell class on Maze-Hard**. Four test puzzles, FPRM above and SFPRM below in each pair, with the conventions of Figure 13. On every maze SFPRM settles walls at ≈90°, free cells that stay empty at ≈113°, free cells that become path at ≈102°, and start and goal at ≈80°, whereas FPRM keeps all classes within 70–87° and separates the two free-cell classes by only a few degrees.

Adaptive (default). $\mathbf{z}_{i+1} = \mathcal{I}_{\eta_i}(\mathbf{z}_i, \widetilde{\mathbf{z}}_i)$ with $\eta_0 = 1$. The step size is multiplied by $\gamma = 0.99$ whenever the residual has not improved for $P = 5$ consecutive steps while it is still above τ. Both models are trained and evaluated with this solver.

Fixed step. The same update with $\eta_i = 1$ throughout, that is, plain fixed-point iteration $\mathbf{z}_{i+1} = \widetilde{\mathbf{z}}_i$.

Anderson acceleration (Anderson, 1965; Walker & Ni, 2011) replaces the latest state by the combination of the last $m + 1$ states whose residuals cancel best. With the columns of $\Delta\mathbf{G}_i$ and $\Delta\widetilde{\mathbf{Z}}_i$ the m differences of consecutive stored residuals and images,

$$\boldsymbol{\omega}_i = \arg\min_{\boldsymbol{\omega}\in\mathbb{R}^m} \|\mathbf{g}_i - \Delta\mathbf{G}_i\boldsymbol{\omega}\|_2^2 + \delta\|\boldsymbol{\omega}\|_2^2, \qquad \mathbf{z}_{i+1} = \Pi\big(\widetilde{\mathbf{z}}_i - \Delta\widetilde{\mathbf{Z}}_i\boldsymbol{\omega}_i\big),$$

with a small ridge $\delta > 0$. The least-squares problem learns, from the recent iterates alone, how the residual responds to a change of state, and the combination removes the components that the plain step overshoots and an oscillation across the fixed point, for example, cancels between consecutive states. We use memory $m = 3$.

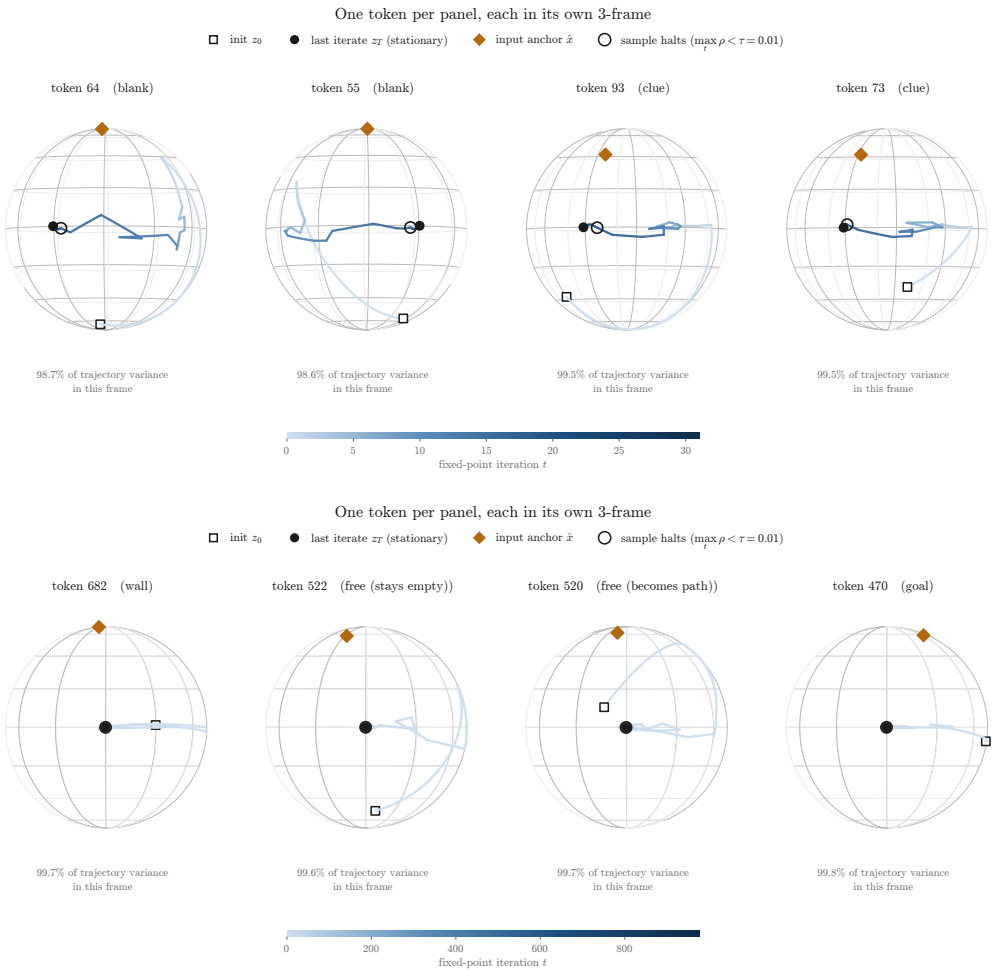


Figure 15: **One SFPRM trajectory per cell class, on the sphere** (Sudoku left, Maze right). For every cell class we draw the token whose 3-frame captures the most trajectory variance. The frame is spanned by the trajectory's two leading singular directions and the anchor x_t, which sits at the pole. x_t is nearly orthogonal to the plane of the trajectory ($|\langle x_t, v_{1,2}\rangle| \approx 0.05$). Arcs are great circles between projected iterates. The colour shows the iteration index.

Broyden's method (Broyden, 1965), the solver commonly used for deep equilibrium models (Bai et al., 2019), takes quasi-Newton steps on $\mathbf{g}(\mathbf{z}) = F_\theta(\mathbf{z}; \mathbf{x}) - \mathbf{z}$ with an estimate $\mathbf{B}_i$ of the inverse Jacobian of $\mathbf{g}$:

$$\mathbf{z}_{i+1} = \Pi\big(\mathbf{z}_i - \mathbf{B}_i \mathbf{g}_i\big), \qquad \mathbf{B}_i = \mathbf{B}_{i-1} + \frac{(\mathbf{s}_i - \mathbf{B}_{i-1}\mathbf{y}_i)\,\mathbf{s}_i^\top \mathbf{B}_{i-1}}{\mathbf{s}_i^\top \mathbf{B}_{i-1}\mathbf{y}_i},$$

where $\mathbf{s}_i = \mathbf{z}_i - \mathbf{z}_{i-1}$ and $\mathbf{y}_i = \mathbf{g}_i - \mathbf{g}_{i-1}$. Starting from $\mathbf{B}_0 = -\mathbf{I}$, the estimate is stored as the last $m = 5$ rank-one corrections.

Neither Anderson's nor Broyden's method computes a Jacobian or any gradient. Both are quasi-Newton methods that infer curvature from secants, the observed changes of state and residual between steps, so each step still costs one evaluation of F_θ. With an empty memory both take exactly the fixed step. Both act on the whole state of an example at once, all tokens flattened into one vector, with one history per example in a batch. Their extrapolated points leave the sphere and are projected back by Π. The extra work per step is a few vector operations on the state and an $m \times m$ linear solve, and the extra memory is m stored copies of the state, both small next to an evaluation of F_θ. As safeguards, a non-finite extrapolation falls back to the fixed step, and Broyden skips an update whose denominator is near zero. Both methods use the step size $\eta = 1$ (no damping).

Table 5: **Inference-time fixed-point solvers on the trained models.** Every example runs 1000 steps from its own random start, shared by the four solvers; each step is one evaluation of F_θ, and an example halts at the first iterate with residual below τ. *Halting*: share of examples that halt (%), median and mean halting step over those, and mean F evaluations over all examples. *Exact at halt*: accuracy over all examples, exact binomial test against the default solver (p), and accuracy among halted examples. *Open loop*: accuracy with halting disabled, at its best step and at $t = 1000$. Maze: all 1000 test mazes; Sudoku: 1000 puzzles evenly spaced over the test set.

	Halting				Exact at halt (%)			Open loop (%)	
Solver	%	med. t	mean t	F evals	all	p	halted	peak (t)	t=1000
SFPRM, Maze (λ=0.575, τ_c=0.01)									
Adaptive (default)	99.2	27	96.3	104.5	87.8	–	87.7	87.8 (20)	87.8
Fixed, η=1	63.1	23	25.8	386.3	87.8	1.00	87.3	87.8 (20)	87.8
Anderson, m=3	100.0	13	13.8	14.8	86.8	0.04	86.8	87.4 (19)	86.7
Broyden, m=5	99.4	14	14.0	20.9	87.4	0.39	87.9	87.9 (16)	51.5
SFPRM, Sudoku (λ=0.1, τ_c=0.01)									
Adaptive (default)	75.0	435	469.9	603.5	95.9	–	100.0	95.9 (933)	95.9
Fixed, η=1	0.0	–	–	1001.0	93.2	<0.001	–	93.2 (973)	93.2
Anderson, m=3	95.4	37	71.3	115.1	95.3	0.34	99.9	95.3 (909)	95.3
Broyden, m=5	89.8	24	49.9	147.8	90.0	<0.001	99.7	69.6 (27)	1.5
FPRM, Maze (τ_{E}=0.1)									
Adaptive (default)	95.9	55	129.3	166.0	86.0	–	86.8	86.1 (102)	86.0
Fixed, η=1	51.4	32	51.9	513.7	85.4	0.07	92.0	85.8 (24)	85.4
Anderson, m=3	100.0	17	18.0	19.0	86.1	1.00	86.1	86.0 (64)	86.0
Broyden, m=5	98.6	16	21.1	35.8	86.0	1.00	87.1	86.1 (19)	32.8
FPRM, Sudoku (τ_{E}=0.1)									
Adaptive (default)	91.5	50	125.0	200.3	88.0	–	96.2	88.1 (972)	88.1
Fixed, η=1	43.8	39	82.6	599.2	80.9	<0.001	100.0	80.9 (932)	80.9
Anderson, m=3	89.2	34	91.1	190.2	88.8	0.38	99.6	89.0 (984)	89.0
Broyden, m=5	18.3	344	389.7	889.3	14.7	<0.001	79.2	13.6 (8)	0.6

C.5.2 RESULTS

We run every solver on all 1000 Maze test examples and on 1000 Sudoku test puzzles spaced evenly over the 422,786-puzzle test set, in fp32, with SFPRM at its trained injection fraction ($\lambda = 0.575$ on Maze, $\lambda = 0.1$ on Sudoku) and FPRM from its released checkpoints. Each example starts from its own random state (uniform on the sphere for SFPRM, the truncated normal of the FPRM solver for FPRM), shared by the four solvers, and is iterated for 1000 steps. We then apply the halting rule to each example separately: the answer is the first iterate whose residual is below τ, and an example that never halts is answered by its iterate at step 1000. Table 5 reports the results, and Figures 5 and 16 show accuracy and residual over the iterations in the four settings.

The undamped step does not halt reliably. The fixed step never reaches τ on 37%, 100%, 49% and 56% of the examples in the four settings. For SFPRM on Maze its answers are identical to the default solver's, but on Sudoku it is less accurate for both models. For SFPRM on Sudoku a quarter of the puzzles do not halt within 1000 steps.

Anderson acceleration matches the default solver at a fraction of the cost. Its accuracy at the halt is within one point of the default solver's in all four settings. Anderson halts earlier in every setting, with a striking 37 against 435 for SFPRM on Sudoku. Averaged over all examples, this is 9× fewer evaluations of F_θ. For FPRM on Sudoku the saving disappears (190 against 200 evaluations): 10.8% of the puzzles never reach τ under Anderson, and all of them are answered incorrectly.

Broyden's method reaches the tolerance threshold quickly but does not stay there. On Maze it halts as early as Anderson (14 and 16 median steps), and its halted answers are as accurate as the

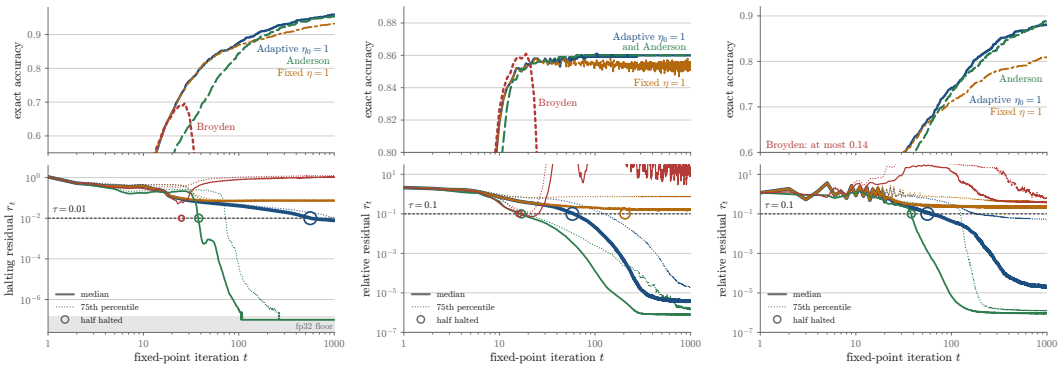


Figure 16: **Accuracy and halting residual over the iterations for the remaining settings of Table 5**. SFPRM on Sudoku ($\lambda = 0.1$, left), FPRM on Maze (middle) and FPRM on Sudoku (right); SFPRM on Maze is Figure 5. Same 1000 examples and solvers as the table. *Top:* exact accuracy of the iterate with halting disabled. *Bottom:* each model's halting residual, median (solid) and 75th percentile (dotted) over the examples, its threshold (dashed), and the iteration by which half of the examples have halted (circle).

default solver's. By continuing iterating beyond the tolerance, however, it leaves the fixed point and its accuracy degrades, making it usable only with halting.

On Sudoku, halting certifies the answer. For SFPRM on Sudoku, 99.7–100% of the puzzles a solver halts on are solved exactly, so the accuracy differences come entirely from the puzzles it fails to halt on. The same holds less strictly for FPRM (96.2% of the default solver's halted puzzles are solved, 99.6% of Anderson's). On Maze, halting does not certify the answer: incorrect mazes converge too, and for the solvers that halt on almost every example the accuracy among halted examples is about the same as the overall accuracy.

Trajectories in the principal plane. Figure 17 shows the four solvers at the trained λ in the plane construction of Figure 4b, on a single test example from each task, from eight initializations. The reference $\mathbf{z}^\star$ is Anderson's end point, a fixed point to fp32 precision that decodes both examples exactly. On Maze, Anderson halts at iteration 13 from all eight initializations, the default solver crosses the tolerance at a median of 140 iterations but ends 0.06 rad away, and fixed $\eta = 1$ never crosses it. On the Sudoku puzzle only Anderson halts (median 40), while the default and fixed solvers end 0.16 and 0.33 rad from $\mathbf{z}^\star$ and still decode the puzzle exactly. Broyden reaches the tolerance as early as Anderson and then leaves from most starts.

Trajectories in polar coordinates. Figures 18 and 19 follow one token per cell class under each solver in the polar view of Figure 5b, for the maze of that panel and for a Sudoku puzzle. The radius is the geodesic angle to a common reference, Anderson's end point after 2000 iterations, a fixed point to fp32 precision that decodes both examples exactly. Inward is convergence toward $\mathbf{z}^\star$, a held radius is orbiting, and outward is leaving. Anderson ends within 10^{-4} degrees of the reference. The default solver ends 0.005–0.14° away on the mazes and up to 2.1° on the Sudoku puzzles, after orbiting for hundreds of iterations; fixed $\eta = 1$ stays up to 0.64° and 4–17° away and never halts; and Broyden reaches the tolerance first and then leaves to 60–103°.

D LIMITATIONS

On the theory side, Theorem 1 certifies a step-size window but not that the patience rule finds it, and Theorem 2 covers only strong injection. On the experimental side, every run uses a single seed and neither model was tuned beyond FPRM's recipe: a broader hyperparameter search, stronger regularization, and optimizers that exploit the spherical geometry further would likely improve both. Two directions remain open. Attention-free MLP-Mixer blocks have been reported to beat Transformer blocks on some of these puzzles (Jolicoeur-Martineau, 2025; Sghaier et al., 2026; Suleymanzade et al., 2026); the spherical loop is agnostic to the block and could be combined with them. And we have not applied SFPRM to language modeling, where looped Transformers are gaining ground

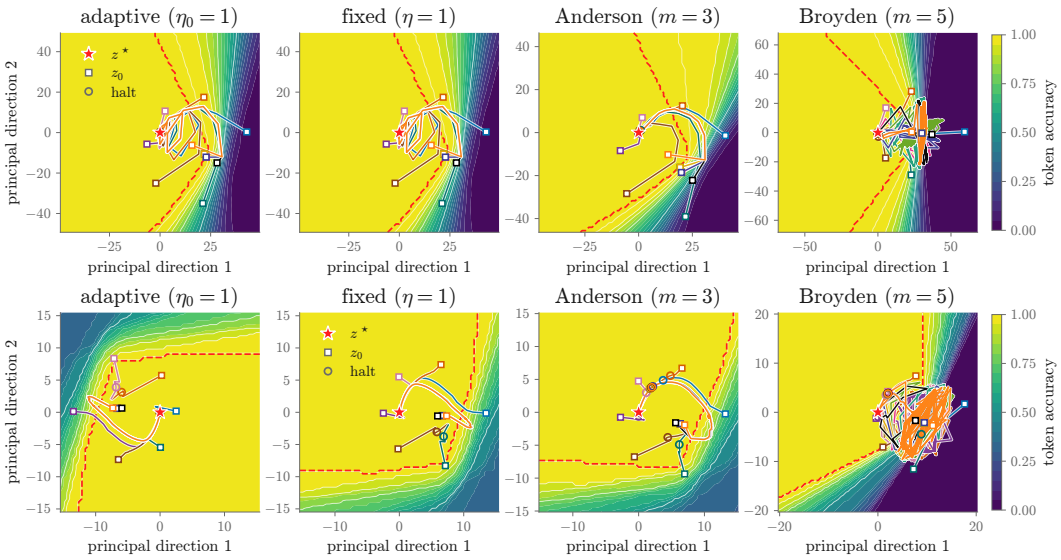


Figure 17: **The four solvers in the principal plane through the fixed point**, for test example 0 of Maze-Hard (top, $\lambda = 0.575$) and of Sudoku-Extreme (bottom, $\lambda = 0.1$). Axes and markers as in Figure 4b, with $\mathbf{z}^\star$ the end point of Anderson acceleration; every solver runs from the same eight initializations (squares) and six further starts placed on its plane. Each panel uses the top-2 principal directions of that solver's own iterates, so the background landscapes differ between panels.

(Geiping et al., 2025; Prairie et al., 2026) and where halting on a fixed angular tolerance may be most valuable.

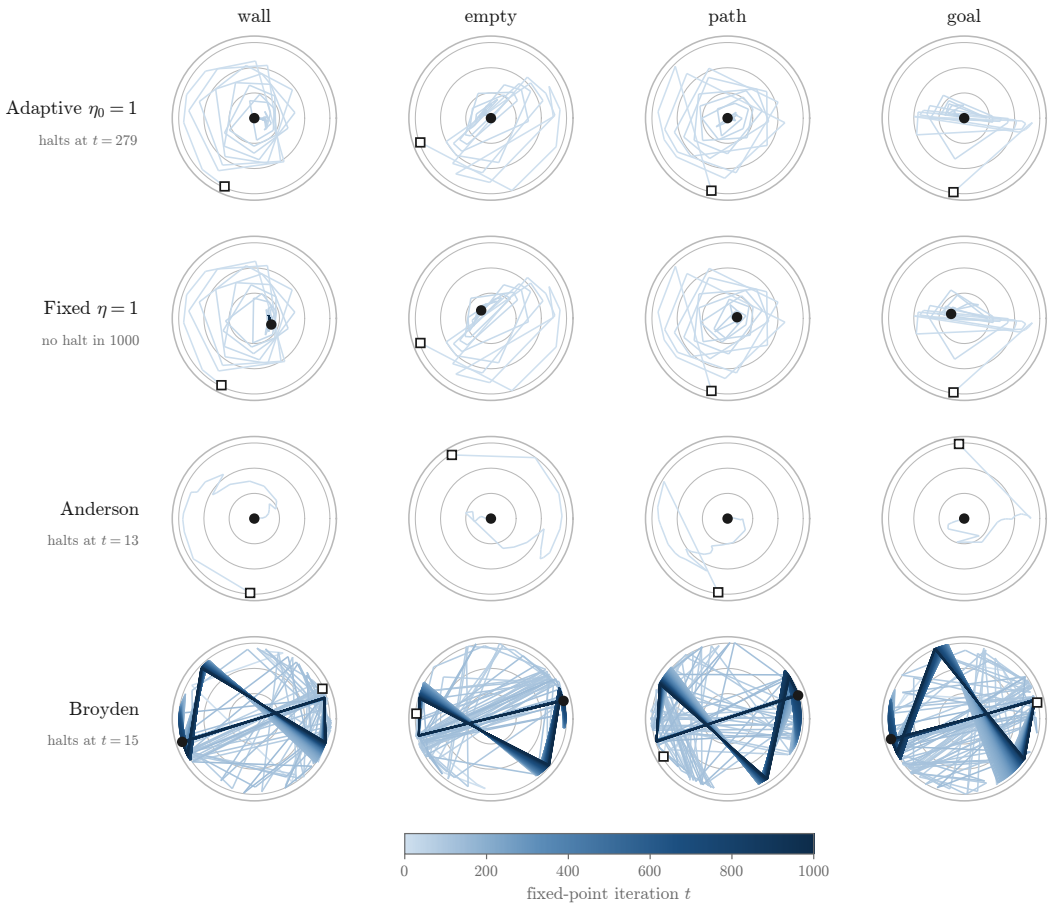


Figure 18: **SFPRM on Maze-Hard: State trajectories in polar coordinates.** The four solvers (rows, with each one's halting step) on the maze of Figure 5b, one token per cell class (columns). Radius: geodesic angle to the reference fixed point, log-scaled from $0.1°$ at the centre to $180°$ at the rim, rings at 1, 10 and $100°$; azimuth: the top-2 principal plane of that token's component orthogonal to the reference; square: start, dot: end of the 1000-iteration run; colour: iteration. Anderson ends at the centre, the default and fixed steps orbit at a small radius, and Broyden leaves for the rim.

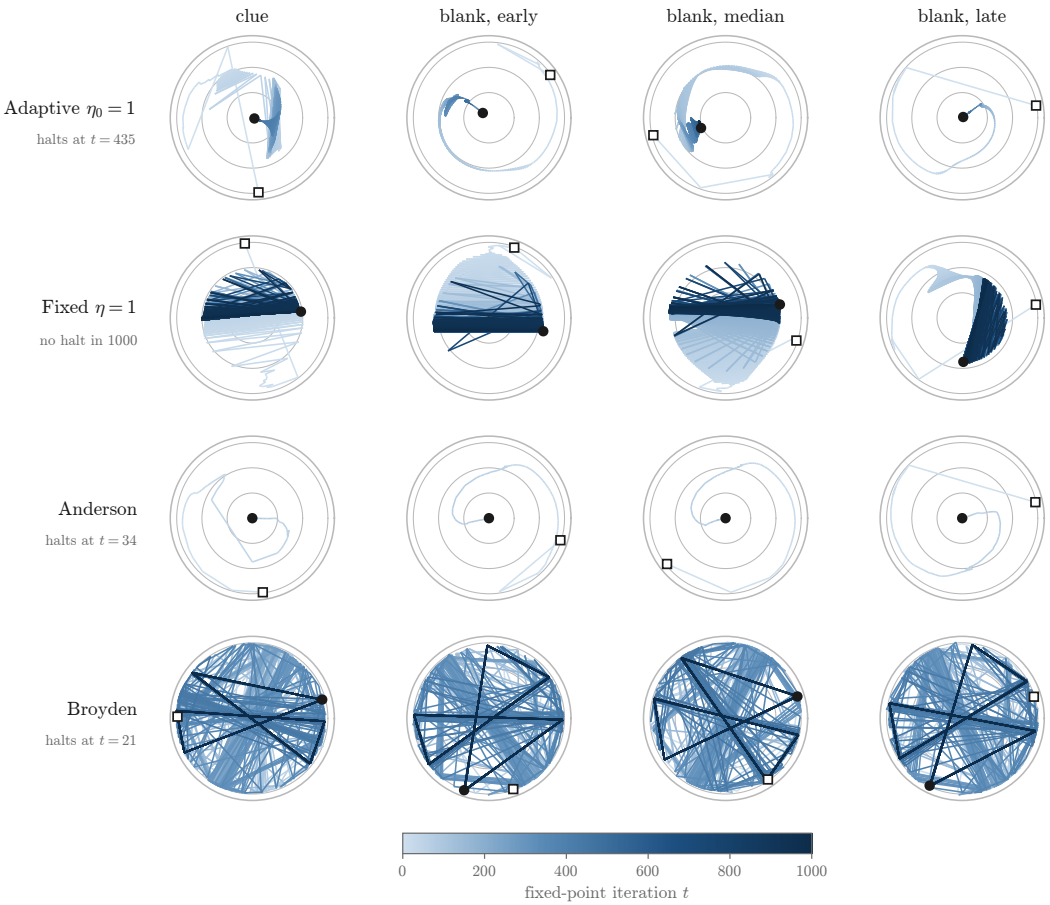


Figure 19: **SFPRM on Sudoku-Extreme: State trajectories in polar coordinates.** A puzzle on which the default solver halts at its median step (435); *columns*: the middle clue cell and the blank cells whose decode settles earliest, at the median and last. Anderson and Broyden reach the tolerance early, the default solver orbits for hundreds of iterations before halting, fixed $\eta = 1$ never halts, and Broyden fills the disc after leaving.

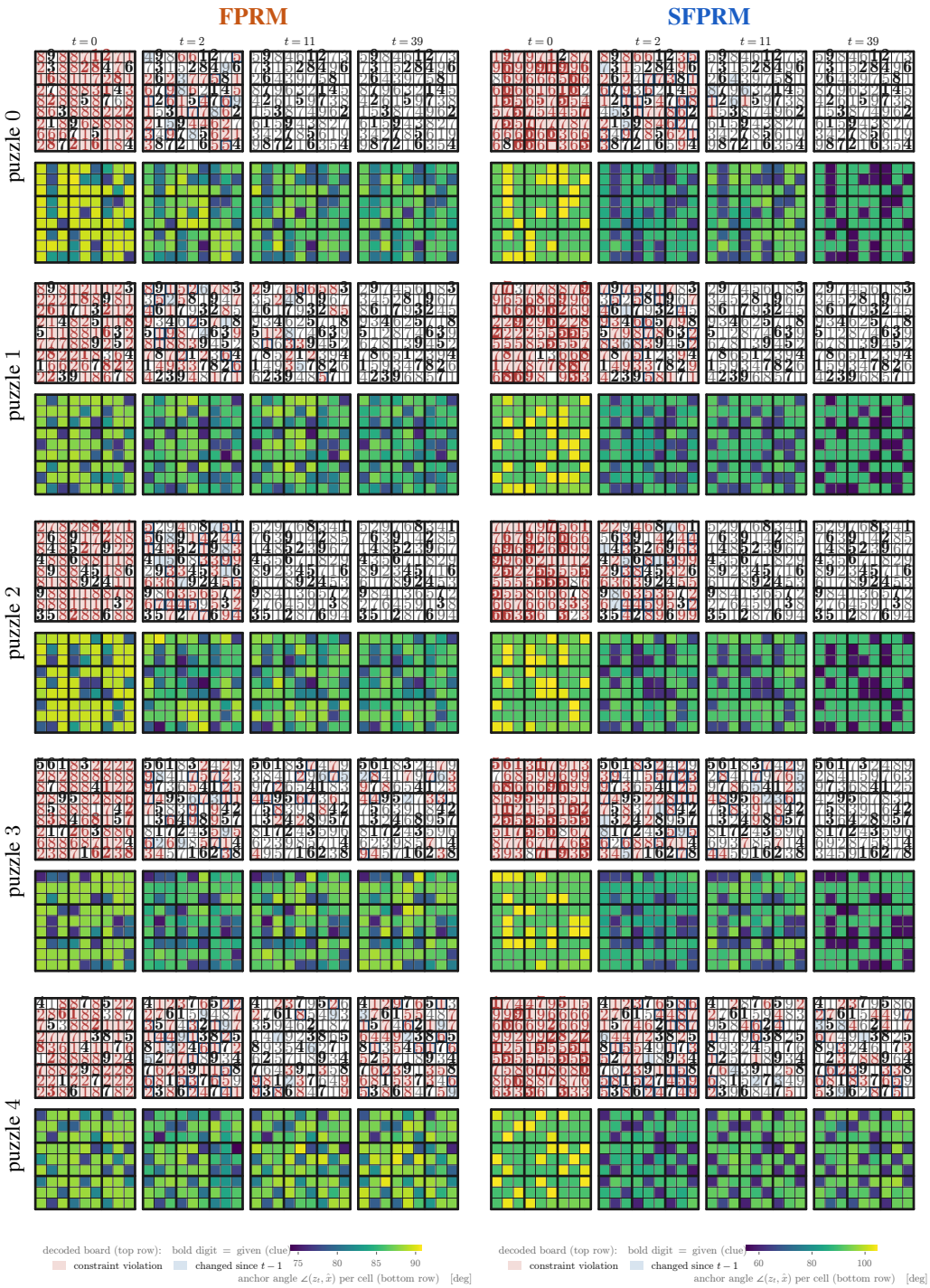


Figure 20: **Filmstrip gallery on Sudoku-Extreme.** Test puzzles 0–4. One test puzzle per row, rolled open-loop for 40 iterations and rendered at $i = 0, 2, 11, 39$ (protocol in Appendix C.4), for FPRM (left) and SFPRM (right): decoded board on top, each cell's angle to its input anchor below.

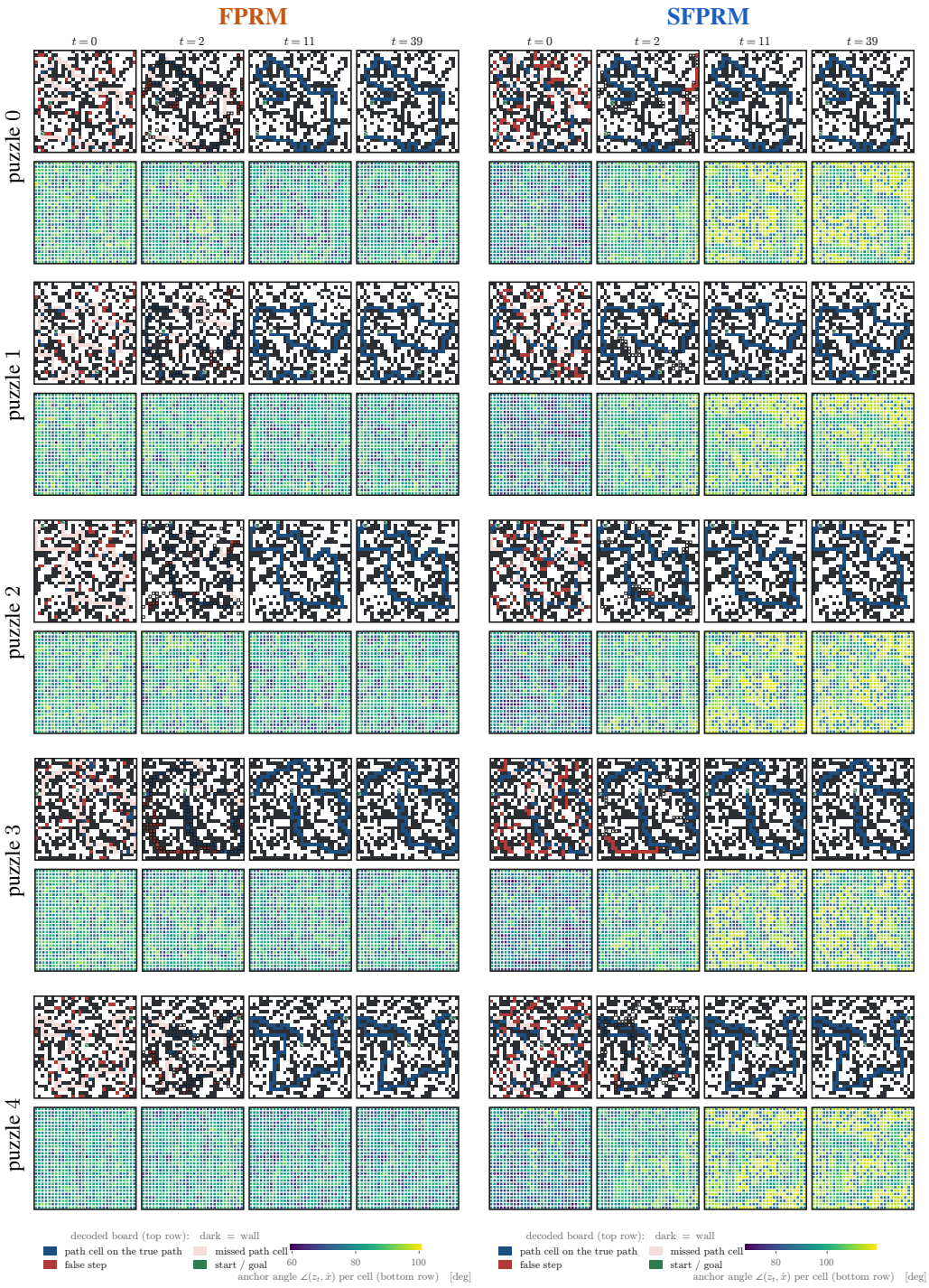


Figure 21: **Filmstrip gallery on Maze-Hard.** Test puzzles 0–4. One 30×30 test maze per row, rolled open-loop for 40 iterations and rendered at $i = 0, 2, 11, 39$ (protocol in Appendix C.4), for FPRM (left) and SFPRM (right): decoded path on top, each cell's angle to its input anchor below.