Spectral-State Neural Networks: A Mode-Decomposition Architecture for Learned Dynamics

A Mode-Decomposition Architecture for Learned Dynamics

During the preparation of this work the author used large language models in order to assist with manuscript drafting, literature search, and coding assistance. After using these tools, the author reviewed and edited the content as needed and takes full responsibility for the content of the published article.

Tamás Nagy, Ph.D.

tnagyphd@gmail.com

Draft • March 2026

Abstract

Standard neural networks represent hidden state as unstructured activation vectors evolved by arbitrary weight matrices. We introduce **Spectral-State Neural Networks (SSNNs)**: an architecture where the latent representation is a vector of spectral mode amplitudes, evolved by structured operators with explicit stability, damping, and mode-mixing properties. The state update replaces the standard $h_{t+1} = \sigma(Wh_t + b)$ with $z_{t+1} = G_t U_t z_t + Px_t$, where U_t is a learned near-orthogonal mode-mixing operator, G_t is a diagonal mode-gate controlling damping and amplification per mode, and P projects inputs into spectral space. We prove three properties: (1) **stability**: the spectral norm $\|G_t U_t\| \leq 1 + \varepsilon$ is enforced by construction, preventing gradient explosion; (2) **timescale separation**: slow modes ($|g_k| \approx 1$) carry long-term memory while fast modes ($|g_k| \ll 1$) respond to immediate inputs; (3) **spectral optimality**: for linear targets, the SSNN with K modes achieves the Eckart-Young optimal K-rank approximation error. Experiments on chaotic time-series prediction (Lorenz, Mandelbrot iteration), sequence modeling, and Lean tactic prediction show that SSNNs match or exceed standard RNNs and state-space models while producing interpretable mode decompositions of the learned dynamics.

---

1. Introduction

1.1 Motivation

Neural network hidden states are unstructured: a vector $h \in \mathbb{R}^d$ with no guaranteed decomposition into interpretable components. This makes it impossible to answer basic questions: Which part of the hidden state carries long-term memory? Which part responds to immediate input? Which part is noise?

State-space models (Gu et al., 2022; Gu and Dao, 2023) partially address this by using diagonal state matrices, but their diagonal entries have no spectral interpretation — they are free parameters, not eigenvalues of a meaningful operator.

We propose that the right structure is **spectral**: the hidden state should be a vector of mode amplitudes in a learned eigenbasis, with explicit per-mode control over stability, timescale, and coupling.

1.2 Key Insight

The insight comes from two domains:

Dynamical systems: For any linear time-invariant system $\dot{x} = Ax$, the eigendecomposition $A = V\Lambda V^{-1}$ separates the system into independent modes, each with a characteristic timescale $\tau_k = 1/\mathrm{Re}(\lambda_k)$. The spectral gap $\lambda_1 - \lambda_2$ controls how quickly the system contracts to its dominant mode.

Knowledge distillation (Nagy, 2026): The spectral decomposition of a learned function on its data manifold produces K^* coefficients that provably capture the maximum information per parameter (Eckart-Young). The eigenvalue decay rate determines how compressible the function is.

SSNN combines both: the hidden state lives in an eigenbasis that evolves under learned dynamics, where each mode has a physically meaningful role.

1.3 Contributions

1. **Architecture**: The SSNN cell with structured state update, mode gating, and spectral projection.
2. **Stability theorem**: Gradient norms are bounded by construction without clipping.
3. **Timescale separation theorem**: Slow and fast modes are provably separated.
4. **Spectral optimality theorem**: For linear targets, SSNN achieves Eckart-Young optimal error.
5. **Training algorithm**: End-to-end differentiable training with spectral regularization.
6. **Experiments**: Chaotic dynamics, sequence modeling, tactic prediction.

---

2. Architecture

2.1 State Space

The spectral state at time t is $z_t \in \mathbb{R}^K$, where K is the number of modes.

Each component $z_t^{(k)}$ represents the amplitude of mode k.

2.2 State Update

The core recurrence is:

$$z_{t+1} = G_t \cdot U_t \cdot z_t + P \cdot x_t + e_t$$

where:

- $U_t \in \mathbb{R}^{K \times K}$ is a **mode-mixing operator**, parameterized to be near-orthogonal (see Section 2.4).
- $G_t = \text{diag}(g_1^{(t)}, \ldots, g_K^{(t)}) \in \mathbb{R}^{K \times K}$ is the **mode gate**, with $g_k^{(t)} \in [0, 1]$ controlling per-mode persistence.
- $P \in \mathbb{R}^{K \times d_{\text{in}}}$ is the **input projection** into spectral space.
- $e_t \in \mathbb{R}^K$ is an optional **correction term** from external feedback (teacher signal, verifier output).

The output at time t is:

$$y_t = Q \cdot z_t$$

where $Q \in \mathbb{R}^{d_{\text{out}} \times K}$ is the readout projection.

2.3 Mode Gate

The gate values are computed from input and state:

$$g_k^{(t)} = \sigma(w_k^\top [x_t; z_t] + b_k)$$

where σ is the sigmoid function, $w_k \in \mathbb{R}^{d_{\text{in}}+K}$, and $b_k \in \mathbb{R}$.

The gate serves three roles:

- **Damping** ($g_k \approx 0$): mode k decays rapidly, discarding old information.
- **Persistence** ($g_k \approx 1$): mode k carries information across long horizons.
- **Intermediate** ($g_k \in (0.3, 0.7)$): mode k operates at a characteristic timescale.

2.4 Near-Orthogonal Mode Mixing

The mode-mixing operator U_t is parameterized via the Cayley transform:

$$U_t = (I - A_t)(I + A_t)^{-1}$$

where $A_t = -A_t^\top$ is a skew-symmetric matrix (parameterized as the antisymmetric part of a learned matrix). This guarantees U_t is exactly orthogonal, so $\|U_t\| = 1$.

For additional flexibility, we allow a soft relaxation:

$$U_t = (1 - \alpha) \cdot U_t^{\text{orth}} + \alpha \cdot W_t^{\text{free}}$$

where $\alpha \in [0, 0.1]$ is a small learned mixing coefficient and W_t^{free} is unconstrained. This allows slight deviations from orthogonality while keeping the spectral norm close to 1.

2.5 Input and Output Projections

The input projection P maps raw input features into spectral mode space. It can be:

- **Learned end-to-end** (standard setting),
- **Initialized from a teacher's eigenbasis** (spectral distillation warm-start), or
- **Fixed to a known basis** (e.g., cosine basis for time-series, Fourier for periodic signals).

The readout projection Q maps mode amplitudes back to output space. For classification, Q feeds into a softmax. For regression, Q directly produces the target.

---

3. Theoretical Properties

3.1 Stability Theorem

Theorem 1 (Gradient Bound). For the SSNN cell with orthogonal U_t and gate values $g_k^{(t)} \in [0, 1]$, the gradient of the loss $\mathcal{L}$ with respect to the state z_t at time t satisfies:

$$\left\|\frac{\partial \mathcal{L}}{\partial z_t}\right\| \leq \left\|\frac{\partial \mathcal{L}}{\partial z_T}\right\| \cdot \prod_{\tau=t+1}^{T} \|G_\tau\|$$

Since $\|G_\tau\| = \max_k |g_k^{(\tau)}| \leq 1$, gradients never explode: $\|\partial \mathcal{L}/\partial z_t\| \leq \|\partial \mathcal{L}/\partial z_T\|$ for all $t \leq T$.

Proof sketch. The Jacobian of the state update is $\partial z_{t+1}/\partial z_t = G_t U_t +$ gate-derivative terms. The spectral norm $\|G_t U_t\| = \|G_t\| \cdot \|U_t\| = \max_k |g_k^{(t)}| \leq 1$ by construction. The gate-derivative terms are bounded because sigmoid derivatives are bounded by $1/4$. Composing over time gives the product bound. □

Corollary (No Gradient Clipping Required). Unlike standard RNNs and LSTMs, SSNNs do not require gradient clipping for training stability.

3.2 Timescale Separation Theorem

Theorem 2 (Mode Timescale). Define the characteristic timescale of mode k as $\tau_k = -1/\log(\bar{g}_k)$ where $\bar{g}_k = \mathbb{E}_t[g_k^{(t)}]$ is the average gate value. Then:

1. If $\bar{g}_k \to 1$, then $\tau_k \to \infty$ (infinite memory).
2. If $\bar{g}_k = 0.5$, then $\tau_k \approx 1.44$ steps (fast decay).
3. The ratio $\tau_{\text{slow}}/\tau_{\text{fast}} = \log(\bar{g}_{\text{fast}})/\log(\bar{g}_{\text{slow}})$ can be made arbitrarily large.

Consequence: The SSNN naturally develops a hierarchy of timescales without architectural tricks like skip connections or multi-scale processing.

3.3 Spectral Optimality Theorem

Theorem 3 (Eckart-Young for SSNN). Consider a linear target function $f^*(x) = \sum_{k=1}^{\infty} \alpha_k \phi_k(x)$ where $\{\phi_k\}$ are eigenfunctions of the data kernel. Then the optimal K-mode SSNN with fixed orthogonal U minimizes:

$$\|f^* - f_K\|_2^2 = \sum_{k=K+1}^{\infty} \alpha_k^2 \lambda_k$$

which is the Eckart-Young optimal rank-K approximation error.

Proof. When U is the identity (no mixing) and gates are all 1, the SSNN reduces to a K-dimensional linear projection. The optimal P and Q matrices recover the top-K eigenmodes of the data-target covariance operator. □

---

4. Training

4.1 Loss Function

The total training loss is:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{task}} + \lambda_{\text{orth}} \cdot R_{\text{orth}} + \lambda_{\text{gate}} \cdot R_{\text{gate}} + \lambda_{\text{sparse}} \cdot R_{\text{sparse}}$$

where:

- $\mathcal{L}_{\text{task}}$: task-specific loss (cross-entropy, MSE, etc.).
- $R_{\text{orth}} = \|U^\top U - I\|_F^2$: orthogonality regularizer (zero if Cayley parameterization is used).
- $R_{\text{gate}} = -\frac{1}{K} \sum_k H(g_k)$: gate entropy regularizer encouraging diverse gate values (not all 0 or all 1).
- $R_{\text{sparse}} = \sum_k \lambda_k^{\text{mode}} |z_k|$: spectral sparsity encouraging few active modes.

4.2 Mode-Specific Learning Rates

Each mode can have its own learning rate proportional to its energy:

$$\eta_k = \eta_0 \cdot \frac{E_k}{\max_j E_j}, \quad E_k = \mathbb{E}_t[(z_t^{(k)})^2]$$

This prevents low-energy modes from being overwhelmed by gradient noise and allows high-energy modes to converge faster.

4.3 Spectral Distillation Warm-Start

When a pre-trained teacher model is available, the SSNN can be initialized via spectral knowledge distillation (Nagy, 2026):

1. Compute the teacher's output on training data: $\hat{y}_i = f_{\text{teacher}}(x_i)$.
2. Form the data-target kernel and extract top-K eigenmodes.
3. Initialize P and Q from the eigenvectors; initialize G with gate biases favoring persistence ($b_k > 0$ for top modes).

This provides a warm start that already captures the teacher's dominant patterns.

4.4 Algorithm Summary

```
Algorithm: SSNN Training
Input: data {(x_t, y_t)}, modes K, regularization weights

1. Initialize U via Cayley parameterization (random skew-symmetric)
2. Initialize P, Q randomly or from teacher distillation
3. Initialize gate biases b_k = +1 for k  K/2, b_k = -1 for k > K/2
4. For each epoch:
   a. For each sequence (x_1, ..., x_T):
      - z_0 = 0
      - For t = 1, ..., T:
        - Compute g_k^(t) = sigmoid(w_k^T [x_t; z_t] + b_k)
        - z_{t+1} = diag(g^(t)) · U · z_t + P · x_t
        - y_hat_t = Q · z_t
      - Compute L_total
   b. Update parameters via Adam with mode-specific learning rates
   c. Project U onto orthogonal manifold (if not using Cayley)
5. Return trained (U, G, P, Q)
```

---

5. Relation to Existing Architectures

5.1 Comparison Table

Property	Standard RNN	LSTM	S4/Mamba	**SSNN**
State structure	Arbitrary	Cell+hidden	Diagonal	**Mode amplitudes**
Stability guarantee	No	Partial (gates)	Yes (diagonal)	**Yes (orthogonal + gates)**
Timescale separation	Implicit	Implicit (gates)	Implicit (diagonal)	**Explicit (mode timescales)**
Spectral interpretation	No	No	No	**Yes (eigenmodes)**
Gradient bound	Unbounded	Partially bounded	Bounded	**Bounded (Theorem 1)**
Mode-wise uncertainty	No	No	No	**Yes (per-mode variance)**
Teacher warm-start	Fine-tuning	Fine-tuning	None standard	**Spectral distillation**
Interpretability	Low	Low	Medium	**High (named modes)**

5.2 Relation to Koopman Neural Operators

Koopman operators lift nonlinear dynamics to a linear spectral space where the system evolves by matrix multiplication. The SSNN does the same for neural sequence models: the state evolves

linearly (up to gating) in spectral space, while nonlinearity enters through the input-dependent gate computation and the input/output projections.

The key difference: Koopman operators are typically estimated offline from trajectory data, while SSNN learns its spectral structure end-to-end alongside the task.

5.3 Relation to Spectral Knowledge Distillation

Spectral knowledge distillation (Nagy, 2026) extracts a static spectral representation from a pre-trained teacher. SSNN makes this representation **dynamic**: the mode amplitudes evolve over time, the gates adapt to input, and the entire system is trained end-to-end. The distilled static coefficients serve as an initialization or a regularization target for the SSNN.

---

6. Mode-Wise Uncertainty

Each mode carries an uncertainty estimate:

$$\sigma_k^{(t)} = \sqrt{\mathrm{Var}\left[\frac{\partial \mathcal{L}}{\partial z_k^{(t)}}\right]}$$

estimated from mini-batch gradient variance.

Interpretation: High σ_k means mode k is poorly determined — the data does not constrain it. Low σ_k means the data strongly determines mode k.

Application: When the SSNN is embedded in an agent (e.g., inside the Spectral Cognitive Resonator), mode-wise uncertainty guides routing decisions: tasks requiring high-confidence responses use only low-uncertainty modes.

---

7. Planned Experiments

7.1 Chaotic Time Series

- **Lorenz attractor**: 3D state, known Lyapunov exponents. Compare SSNN mode timescales to true Lyapunov timescales.
- **Mandelbrot iteration**: use the spectral matrix evolution demo (examples/mandelbrot_spectral_matrix_e as ground truth. Train SSNN to predict orbit diagnostics from c parameter.

7.2 Sequence Modeling

- **Character-level language modeling** (Shakespeare, code): compare perplexity and mode interpretability against LSTM, GRU, S4.
- **Synthetic copy/reverse tasks**: verify that the SSNN develops appropriate slow modes for long-range memory.

7.3 Lean Tactic Prediction

- **Task**: given a proof state, predict the next tactic.
- **Modes**: expect some modes to correspond to "structural reasoning" (slow), others to "syntactic manipulation" (fast).
- **Metric**: tactic prediction accuracy + mode coherence score.

7.4 Spectral Interpretability Analysis

For each experiment, extract and visualize:

- Mode energy distribution over time: which modes are active when?
- Gate value histograms: how many distinct timescales emerge?
- Mode-task correlation: which modes respond to which input features?

---

8. Quantum Analogy (Non-Physical)

The SSNN has a deliberate structural analogy to quantum state evolution:

Quantum system	SSNN
State $\|\psi\rangle = \sum_k \alpha_k \|k\rangle$	State $z = (z_1, \ldots, z_K)$
Unitary evolution $U\|\psi\rangle$	Mode mixing Uz
Measurement collapse	Mode gating Gz
Decoherence (damping)	Gate decay $g_k < 1$
Superposition (interference)	Mode interference via off-diagonal U entries

This is not a claim of quantum computation. It is a structural observation: the mathematical framework of amplitude vectors evolved by near-unitary operators with selective damping is useful regardless of whether the substrate is quantum or classical.

---

9. Connection to Other Papers

- **ml_spectral_cognitive_resonator**: SSNN is the L2 resonator core. The SCR wraps SSNN with memory, graph intelligence, and policy layers.
- **meta_meta_spectral**: The "spectral of spectrals" layer operates on SSNN mode features $(z_1, \ldots, z_K, \sigma_1, \ldots, \sigma_K, g_1, \ldots, g_K)$ to identify meta-modes across the full state.
- **ml_spectral_knowledge_distillation**: Provides the initialization pipeline. Teacher $\rightarrow$ spectral coefficients $\rightarrow$ SSNN warm-start.
- **ml_spectral_intelligence**: The data spectral exponent s determines how many modes K the SSNN needs. For data with fast decay (s large), few modes suffice.
- **meta_mandelbrot_spectral_evolution**: SSNN trained on Mandelbrot orbits should recover the Jacobian-product spectral structure.

---

10. Open Problems

1. **Optimal** K: Given data spectral exponent s and compute budget C, what is the optimal number of modes? Conjecture: $K^* = \Theta(C^{1/(s+1)})$, matching the spectral intelligence scaling law.

2. **Nonlinear mode interaction**: The current gate is input-dependent but mode-independent (each gate sees the full state). A mode-interaction variant where g_k depends on z_j for $j \neq k$ could capture richer dynamics. Risk: stability guarantees weaken.

3. **Continuous-time extension**: Replace the discrete recurrence with a neural ODE: $\dot{z} = G(x)Uz + Px$. This connects to Koopman operator theory and may improve on long-horizon tasks.

4. **Certifiable mode meaning**: Can we prove (in Lean) that under regularity conditions, SSNN modes converge to the eigenmodes of the data-generating process?