Learning Interpretable Dynamics of Temporal Networks via Neural ODEs and Symbolic Regression

Connor Smith[1] and Giulio V. Dalla Riva[1]

[1]School of Mathematics and Statistics, University of Canterbury, New Zealand

Abstract

Temporal networks—networks whose structure changes over time—appear across domains from neuroscience to ecology to social systems. While most approaches focus on predicting future network states, they rarely provide interpretable models of the underlying dynamics. We present a framework that learns continuous, interpretable differential equations governing the evolution of temporal network structure. Our approach embeds networks into a low-dimensional latent space via Random Dot Product Graphs (RDPG), learns the dynamics of this embedding using Neural Ordinary Differential Equations (Neural ODEs), and extracts human-interpretable equations through symbolic regression. We develop a gauge-theoretic analysis showing that RDPG embeddings have rotational ambiguity, and derive a gauge-consistent architecture $\dot{X} = N(P)X$ with symmetric N that eliminates this ambiguity while achieving dramatic parameter reduction (from ≈10,000 to as few as 2 parameters). We demonstrate the framework on synthetic temporal networks, showing that it successfully recovers governing equations and dynamical parameters. This work bridges the gap between predictive accuracy and mechanistic understanding in temporal network modeling.

1 Introduction

Temporal networks—networks whose edges and nodes change over time—are ubiquitous in complex systems [1]. Examples include protein interaction networks that rewire during cellular processes [2], social networks where relationships form and dissolve [3], and ecological networks whose structure responds to environmental change [4]. Understanding how and why network structure changes is central to predicting system behavior.

Most temporal network modeling falls into two categories. The first models *dynamics on networks*: how node states evolve given a fixed or slowly-changing network topology (e.g., epidemic spreading, opinion dynamics) [5]. The second models *dynamics of networks*: how the network structure itself evolves [6]. This paper addresses the latter, which remains less developed despite its importance.

Existing approaches to modeling network dynamics face a fundamental tension. Statistical models like temporal exponential random graphs [3] are interpretable but often lack predictive power. Machine learning approaches [7] achieve better predictions but function as black boxes, offering little insight into the mechanisms driving structural change. We propose a framework that achieves both: predictive models that can be distilled into interpretable differential equations.

Our key insight is that the discreteness of network events (edges appearing or disappearing) can be overcome by working in a continuous embedding space. We use Random Dot Product Graphs (RDPG) [8] to embed each network snapshot into a low-dimensional latent space where similar nodes cluster together and connection probabilities arise naturally from inner products. The temporal evolution of these embeddings is then smooth and amenable to differential equation modeling.

We train Neural Ordinary Differential Equations (Neural ODEs) [9] to learn the dynamics in embedding space. While Neural ODEs provide excellent fits, they remain opaque. We therefore apply symbolic regression to discover closed-form differential equations that approximate the learned neural dynamics. These equations are interpretable—they can be analyzed mathematically, checked for conservation laws, and compared across systems.

Contributions. We introduce:
1. A complete pipeline from temporal network observations to interpretable differential equations
2. A gauge-theoretic analysis of RDPG dynamics, identifying what can and cannot be learned from embedding trajectories
3. Parsimonious architectures ($\dot{X} = N(P)X$ with symmetric N) that are gauge-consistent by construction and can recover exact dynamical parameters
4. Demonstration on synthetic systems with known ground-truth dynamics
5. Open-source Julia implementation (`RDPGDynamics.jl`) for reproducibility

2 Methods

Our framework consists of three stages: (1) embedding temporal networks via RDPG, (2) learning dynamics with Neural ODEs, and (3) extracting interpretable equations through symbolic regression.

2.1 Random Dot Product Graph Embedding

Given a temporal network represented as a sequence of adjacency matrices $\{A_t\}_{t=1}^{T}$, we embed each snapshot into a latent space using Random Dot Product Graphs (RDPG) [8].

For an adjacency matrix $A \in \{0,1\}^{n \times n}$, the RDPG embedding computes:

$$A \approx \hat{L}\hat{R}^{\top}$$

where $\hat{L}, \hat{R} \in \mathbb{R}^{n \times d}$ are the left and right embedding matrices and $d \ll n$ is the embedding dimension. These are obtained via truncated singular value decomposition (SVD):

$$A = U\Sigma V^{\top} \Rightarrow \hat{L} = U_d \Sigma_d^{\frac{1}{2}}, \quad \hat{R} = V_d \Sigma_d^{\frac{1}{2}}$$

where subscript d denotes truncation to the top d singular values/vectors.

The matrix $P = \hat{L}\hat{R}^{\top}$ has entries $P_{ij} \in [0,1]$ representing the probability of an edge between nodes i and j. This probabilistic interpretation is central to our approach: while edge events are discrete, connection probabilities evolve continuously.

Temporal alignment. SVD decompositions are unique only up to orthogonal transformations. To ensure smooth trajectories across time, we align each embedding to its predecessor using orthogonal Procrustes rotation:

$$\Omega_t = \arg\min_{\Omega^\top \Omega = I} \|\Omega \hat{L}_t - \hat{L}_{t-1}\|_F^2$$

with solution $\Omega_t = VU^\top$ where $\hat{L}_t \hat{L}_{t-1}^\top = U\Sigma V^\top$.

2.2 Dynamics in Latent Space and Gauge Freedom

Our goal is to learn how latent positions evolve over time. We model this evolution as a dynamical system governed by a vector field f:

$$\dot{X} = f(X)$$

where $X \in \mathbb{R}^{n\times d}$ collects all node positions and $f : \mathbb{R}^{n\times d} \to \mathbb{R}^{n\times d}$ specifies the velocity of each node as a function of the current configuration.

Since we observe networks (and hence probability matrices P), not latent positions directly, we need to understand what dynamics on X imply for P. The probability matrix is $P = XX^\top$, so by the product rule:

$$\dot{P} = \dot{X}X^\top + X\dot{X}^\top = f(X)X^\top + Xf(X)^\top$$

This is the **induced dynamics** on the observable P. Any vector field f on latent space induces a corresponding evolution of the probability matrix.

However, the latent positions X are not uniquely determined by P. For any orthogonal matrix $Q \in O(d)$:

$$(XQ)(XQ)^\top = XQQ^\top X^\top = XX^\top = P$$

Thus X and XQ represent the *same* observable P. This is the **gauge freedom** of RDPG: the equivalence class $[X] = \{XQ : Q \in O(d)\}$ corresponds to a single probability matrix.

This gauge freedom has profound implications for learning dynamics. If two configurations X and XQ are observationally indistinguishable, then any dynamics that moves along this equivalence class—rotating all positions by a common orthogonal transformation—produces no observable change in the network. We call such dynamics **invisible**.

Definition 1 (Observable vs. Invisible Dynamics). A vector field $f : \mathbb{R}^{n\times d} \to \mathbb{R}^{n\times d}$ produces *observable dynamics* if $\dot{P} \neq 0$, where $\dot{P} = f(X)X^\top + Xf(X)^\top$. Otherwise, the dynamics are *invisible*—the parameterization changes but the graph structure is static.

Theorem 1 (Characterization of Invisible Dynamics). Let $X \in \mathbb{R}^{n\times d}$ have full column rank. A vector field f produces invisible dynamics if and only if $f(X) = XA$ for some skew-symmetric matrix $A \in \mathfrak{so}(d)$, i.e., $A^\top = -A$.

Proof. ($\Leftarrow$) Suppose $f(X) = XA$ with $A^\top = -A$. Then:

$$\dot{P} = XAX^\top + X(XA)^\top = XAX^\top + XA^\top X^\top = X(A + A^\top)X^\top = 0$$

($\Rightarrow$) Suppose $\dot{P} = f(X)X^\top + Xf(X)^\top = 0$.

Decompose $f(X) = XA + W$ where $A = (X^\top X)^{-1} X^\top f(X)$ and $X^\top W = 0$ (i.e., W lies in the orthogonal complement of $\mathrm{col}(X)$).

Substituting into $\dot{P} = 0$:

$$0 = (XA + W)X^\top + X(XA + W)^\top = X(A + A^\top)X^\top + WX^\top + XW^\top$$

For X with full column rank, consider the constraint $X^\top W = 0$ combined with the equation above. Taking the projection onto $\mathrm{col}(X)$: multiplying on left by $(X^\top X)^{-1} X^\top$ and on right by $X(X^\top X)^{-1}$:

$$0 = A + A^\top + (X^\top X)^{-1} X^\top W X^\top X (X^\top X)^{-1} + \text{similar term}$$

Since $X^\top W = 0$, the middle terms vanish, giving $A + A^\top = 0$.

For the remaining equation $WX^\top + XW^\top = 0$ with $X^\top W = 0$: this is a symmetric matrix equation. For generic full-rank X, the only solution is $W = 0$. $\square$

Corollary 1 The space of invisible dynamics is isomorphic to $\mathfrak{so}(d)$, with dimension $\frac{d(d-1)}{2}$. For $d = 2$, this is 1-dimensional (a single rotation rate); for $d = 3$, it is 3-dimensional.

The invisible dynamics $\dot{X} = XA$ are infinitesimal rotations along gauge orbits. In particular, *uniform rotation around the origin* satisfies $\dot{X}_i = X_i A$, producing $\dot{P} = 0$—the embedding rotates but the network is static.

Crucially, other rotational dynamics *are* observable. Rotation around the *origin* is gauge (invisible), but circulation around a *nonzero centroid* is observable:

Proposition 1 (Centroid Circulation). Dynamics of the form $\dot{X}_i = (X_i - |(X))A$ with $|(X) \neq 0$ and $A \in \mathfrak{so}(d)$ produce observable changes in P.

Proof. Let $|(X) = \frac{1}{n} \sum_i X_i$ be the centroid. The dynamics $\dot{X}_i = (X_i - |(X))A$ can be rewritten as:

$$\dot{X}_i = X_i A - |(X)A$$

In matrix form with $\mathbf{1} \in \mathbb{R}^n$ the all-ones vector:

$$\dot{X} = XA - \mathbf{1}|(X)^\top A$$

Computing $\dot{P}$:

$$\begin{aligned}
\dot{P} &= \dot{X}X^\top + X\dot{X}^\top \\
&= (XA - \mathbf{1}|(X)^\top A)X^\top + X(XA - \mathbf{1}|(X)^\top A)^\top \\
&= XAX^\top + XA^\top X^\top - \mathbf{1}|(X)^\top AX^\top - XA^\top\ |(X)\mathbf{1}^\top
\end{aligned}$$

The first two terms cancel since $A + A^\top = 0$. Let $v = A^\top\ |(X) = -A\ |(X)$:

$$\dot{P} = \mathbf{1}v^\top X^\top + Xv\mathbf{1}^\top$$

Entry-wise: $\dot{P}_{ij} = v \cdot X_j + X_i \cdot v$.

This vanishes for all i, j only if $v = 0$, i.e., $A\,|(X) = 0$. For generic $|(X) \neq 0$ and $A \neq 0$, we have $\dot{P} \neq 0$. ☐

Interpretation. Circulation around the centroid decomposes as:

$$\dot{X}_i = \underbrace{X_i A}_{\text{invisible (gauge)}} - \underbrace{|(X)A}_{\text{shared drift (observable)}}$$

The first term is pure gauge. The second is a constant velocity applied to all nodes, which shifts all dot products and hence changes P.

Similarly, differential rotation rates are observable:

Proposition 2 (Differential Rotation is Observable). If nodes have different rotation rates:

$$\dot{X}_i = X_i A_i, \quad A_i \in \mathfrak{so}(d)$$

then:

$$\dot{P}_{ij} = X_i (A_i - A_j) X_j^\top$$

This is generically nonzero when $A_i \neq A_j$.

Proof.

$$\begin{aligned}
\dot{P}_{ij} &= \dot{X}_i X_j^\top + X_i \dot{X}_j^\top = (X_i A_i) X_j^\top + X_i \left(X_j A_j\right)^\top \\
&= X_i A_i X_j^\top + X_i A_j^\top X_j^\top = X_i A_i X_j^\top - X_i A_j X_j^\top \\
&= X_i (A_i - A_j) X_j^\top
\end{aligned}$$

using $A_j^\top = -A_j$. This is nonzero when $A_i \neq A_j$ and X_i, X_j are generic. ☐

Dynamics	$\dot{P} = 0$?	**Observable?**
$\dot{X} = XA$ (uniform rotation around origin)	Yes	No
$\dot{X}_i = (X_i - \|(X))A$ with $\|(X) \neq 0$	No	Yes
$\dot{X}_i = X_i A_i$ with $A_i \neq A_j$	No	Yes
$\dot{X}_i = \alpha X_i$ (radial scaling)	No	Yes
$\dot{X}_i = \sum_j w_{ij} (X_j - X_i)$ (attraction/repulsion)	No	Yes

Table 1: Classification of dynamics by observability.

A natural question arises: given that some dynamics are invisible, can we still learn the observable part?

Theorem 2 (Identifiability Modulo Gauge). Let $X \in \mathbb{R}^{n \times d}$ have full column rank. Given $\dot{P}$ and X, the vector field $f(X)$ is uniquely determined up to gauge:

$$f(X) = F + XA$$

where F is any solution to $\dot{P} = FX^\top + XF^\top$ and $A \in \mathfrak{so}(d)$ is arbitrary.

Theorem 3 (Gauge Equivalence). Two vector fields f and $\tilde{f}$ are gauge equivalent (induce the same $\dot{P}$) if and only if:

$$f(X) - \tilde{f}(X) = XA(X)$$

for some $\mathfrak{so}(d)$-valued function $A(X)$.

Proof. Apply Theorem 1 to the difference $h = f - \tilde{f}$. □

Corollary 2 (Canonical Decomposition). Any vector field decomposes uniquely as:

$$f(X) = f_{\text{phys}}(X) + XA(X)$$

where f_{phys} determines $\dot{P}$ and XA is pure gauge.

This decomposition clarifies what can be learned: the "physical" content—what affects the observable—is uniquely determined; only the coordinate-dependent form varies with gauge choice.

Gauge-Free Decomposition of $\dot{X}$. Any velocity $\dot{X}$ decomposes uniquely as $\dot{X} = XA + W$ where $A = \left(X^\top X\right)^{-1} X^\top \dot{X} \in \mathbb{R}^{d \times d}$ and $W \perp \text{col}(X)$. Further decomposing $A = A_{\text{sym}} + A_{\text{skew}}$:

Component	Contributes to $\dot{P}$?	Interpretation
XA_{sym}	Yes	Radial/stretching dynamics
XA_{skew}	No	Pure rotation (gauge)
W	Yes	Rotates $\text{col}(P)$ in $\mathbb{R}^n$

Table 2: Decomposition of $\dot{X}$ into observable and gauge components.

The observable content of $\dot{X}$ is $\left(A_{\text{sym}}, W\right)$.

Implications for learning. When we train on estimated positions $\widehat{X}(t)$, the Procrustes alignment fixes a consistent gauge. The learned f determines $\dot{P}$ correctly, but a different alignment procedure would yield a gauge-equivalent $f + XA$.

2.3 Neural ODE Dynamics

After embedding, we have a sequence of latent positions $\left\{\hat{L}_t\right\}_{t=1}^{T}$. We flatten each $\hat{L}_t \in \mathbb{R}^{n \times d}$ into a vector $\boldsymbol{u}_t \in \mathbb{R}^{nd}$ and model the dynamics as:

$$\frac{d\boldsymbol{u}}{dt} = f_{\theta(\boldsymbol{u})}$$

where f_θ is a neural network with parameters θ.

We parameterize f_θ as a fully-connected network with architecture:

$$f_\theta : \mathbb{R}^{nd} \xrightarrow{\text{Dense}} \mathbb{R}^{128} \xrightarrow{\text{celu}} \mathbb{R}^{128} \xrightarrow{\text{celu}} \mathbb{R}^{64} \xrightarrow{\text{celu}} \mathbb{R}^{nd}$$

Training minimizes the prediction error:

$$\mathcal{L}(\theta) = \sum_{t=1}^{T} \|\boldsymbol{u}_t - \hat{\boldsymbol{u}}_{t(\theta)}\|_2^2 + \lambda \mathcal{L}_{\text{prob}}$$

where $\hat{\boldsymbol{u}}_t$ is obtained by integrating the Neural ODE from $\boldsymbol{u}_1$, and $\mathcal{L}_{\text{prob}}$ penalizes predicted probabilities outside $[0, 1]$:

$$\mathcal{L}_{\text{prob}} = \sum_{i \neq j} \max\left(0, -P_{ij}\right) + \max\left(0, P_{ij} - 1\right)$$

We use a two-stage optimization: Adam for initial exploration followed by Lion for fine-tuning. Gradients are computed via adjoint sensitivity analysis for memory efficiency [9].

2.4 Universal Differential Equations

When domain knowledge suggests a particular functional form for the dynamics, we can incorporate it via Universal Differential Equations (UDEs) [10]. The vector field decomposes as:

$$f(\boldsymbol{u}) = f_{\text{known}}(\boldsymbol{u}; \varphi) + f_{\text{NN}}(\boldsymbol{u}; \theta)$$

where f_{known} encodes known physics with parameters φ, and f_{NN} is a neural network that learns residual corrections.

For RDPG dynamics, gauge theory (Section 2.2) suggests a particularly elegant form.

Theorem 4 (Equivariant Dynamics). Let $X \in \mathbb{R}^{n \times d}$ have full column rank. Any $O(d)$-equivariant vector field $f : \mathbb{R}^{n \times d} \to \mathbb{R}^{n \times d}$ has the form:

$$f(X) = N(P) \cdot X$$

where $N : \mathbb{R}^{n \times n} \to \mathbb{R}^{n \times n}$ depends only on $P = XX^\top$.

Proof. Define $N(X) := f(X)X^\dagger$ where $X^\dagger = \left(X^\top X\right)^{-1} X^\top$ is the Moore-Penrose pseudoinverse.

For equivariance, we require $f(XQ) = f(X)Q$ for all $Q \in O(d)$. Then:

$$N(XQ) = f(XQ)(XQ)^\dagger = f(X)Q\left(Q^\top X^\dagger\right) = f(X)X^\dagger = N(X)$$

So N is constant on $O(d)$-orbits. Since orbits are indexed by $P = XX^\top$ (two matrices $X, \widetilde{X}$ lie on the same orbit iff $XX^\top = \widetilde{X}\widetilde{X}^\top$), we have $N = N(P)$. $\square$

This form is automatically gauge-consistent since N depends on the observable P, not the gauge-dependent X. The key question is: how should we constrain N to eliminate gauge freedom?

Theorem 5 (Gauge Dynamics are Not Symmetric). The invisible (gauge) dynamics $\dot{X} = XA$ with $A \in \mathfrak{so}(d)$ correspond to $N = XAX^\dagger$. For generic full-rank X and nonzero A, this N is **not symmetric**.

Proof. For $X = I_d$ (taking $n = d$), we have $N = A$, which is skew-symmetric.

For general X with thin SVD $X = U\Sigma V^\top$, we get:

$$N = XAX^\dagger = U\Sigma V^\top AV\Sigma^{-1}U^\top = UBU^\top$$

where $B = \Sigma(V^\top AV)\Sigma^{-1}$.

Since $V^\top AV$ is skew-symmetric (as A is) and Σ generically has distinct singular values, the conjugation by Σ and Σ^{-1} breaks the skew-symmetry: $B \neq B^\top$ unless $A = 0$.

Thus $N = UBU^\top$ is neither symmetric nor skew-symmetric for generic X and nonzero A. □

This theorem is the key insight: *gauge directions correspond to non-symmetric* N. Therefore, constraining N to be symmetric eliminates gauge:

Theorem 6 (Gauge Elimination via Symmetry). Constraining $N(P) = N(P)^\top$ (symmetric) eliminates all non-trivial gauge freedom. Any symmetric N with $NX \neq 0$ produces observable dynamics ($\dot{P} \neq 0$). Moreover, symmetric N can produce *any* realizable $\dot{P}$—no expressivity is lost.

Proof. Suppose $N = N^\top$ and $\dot{P} = NP + PN = 0$.

Let $P = V\Lambda V^\top$ be the spectral decomposition with $\Lambda = \operatorname{diag}(\lambda_1, ..., \lambda_d, 0, ..., 0)$ and $\lambda_i > 0$ for $i \leq d$.

Define $\widetilde{N} = V^\top NV$ (symmetric since N is). The condition $NP + PN = 0$ transforms to:

$$\widetilde{N}\Lambda + \Lambda\widetilde{N} = 0$$

Entry-wise: $\widetilde{N}_{ij}(\lambda_i + \lambda_j) = 0$.

Case analysis:

- For $i, j \leq d$: $\lambda_i + \lambda_j > 0$, so $\widetilde{N}_{ij} = 0$.
- For $i \leq d$, $j > d$: $\lambda_i + 0 = \lambda_i > 0$, so $\widetilde{N}_{ij} = 0$.
- For $i > d$, $j \leq d$: by symmetry $\widetilde{N}_{ij} = \widetilde{N}_{ji} = 0$.
- For $i, j > d$: the constraint $0 \cdot \widetilde{N}_{ij} = 0$ is vacuous.

Therefore $\widetilde{N} = \begin{pmatrix} 0 & 0 \\ 0 & \widetilde{N}_{22} \end{pmatrix}$ where $\widetilde{N}_{22} \in \mathbb{R}^{(n-d)\times(n-d)}$ is arbitrary symmetric (supported on null(P)).

Since col(X) = col(V_1) where V_1 comprises the first d columns of V, we can write $X = V_1 R$ for invertible R. Then:

$$NX = V\widetilde{N}V^\top V_1 R = V\widetilde{N}\begin{pmatrix} I_d \\ 0 \end{pmatrix}R = V\begin{pmatrix} 0 \\ 0 \end{pmatrix} = 0$$

Contrapositive: If $N = N^\top$ and $NX \neq 0$, then $NP + PN \neq 0$, so $\dot{P} \neq 0$. □

This suggests a principled UDE architecture:

$$N(P) = N_{\text{known}}(P) + N_{\text{NN}}(P), \quad \text{both symmetric}$$

where N_{known} might be a polynomial in P (encoding local neighbor influence) and N_{NN} learns corrections.

We consider three parameterization classes in order of increasing expressivity:

Polynomial $N(P)$. The most parsimonious form:

$$N = \alpha_0 I + \alpha_1 P + \alpha_2 P^2 + ... + \alpha_k P^k$$

with only $k + 1$ learnable scalars. The interpretation is intuitive: $\alpha_0 I$ represents intrinsic node dynamics, $\alpha_1 P$ captures direct neighbor influence (one-hop interactions), and $\alpha_2 P^2$ captures two-hop effects through shared neighbors. For many network dynamics, degree $k \leq 2$ suffices.

Gauge invariance of learned parameters. A key advantage of expressing dynamics in terms of P rather than X directly: the scalar coefficients $\alpha_0, \alpha_1, ...$ are *gauge-invariant*. Since $P = XX^\top$ is unchanged by orthogonal transformations $X \mapsto XQ$, and $N(P)$ depends only on P, the learned α_k values are independent of the coordinate system chosen by SVD and Procrustes alignment. This means coefficients learned in *any* gauge (including the DUASE-estimated coordinates) can be applied to *any other* gauge (including the true positions)—the "learn anywhere, apply everywhere" principle. By contrast, dynamics expressed directly in X coordinates (e.g., $\dot{X}_1 = aX_1 + bX_2$) would have coefficients that depend on the specific basis chosen.

Pairwise kernel $N(P)$. A flexible homogeneous form:

$$N_{ij} = \begin{cases} \kappa(P_{ij}) i \neq j \\ h(P_{ii}) \, i = j \end{cases}$$

where $\kappa, h : [0, 1] \to \mathbb{R}$ are learned functions (small neural networks or parametric forms). Symmetry is automatic since $P_{ij} = P_{ji}$. The kernel κ can capture nonlinear responses to connection probability, such as threshold effects or saturation.

General symmetric $N(P)$. A neural network that outputs the upper triangle of a symmetric matrix:

$$\text{NN} : \text{uptri}(P) \mapsto \text{uptri}(N)$$

with $\frac{n(n+1)}{2}$ inputs and outputs. This is the most expressive but least parsimonious option.

Table 3 summarizes the parameter counts.

Architecture	**Parameters**	**Expressivity**
Polynomial ($k = 1$)	2	Low
Polynomial ($k = 2$)	3	Low
Pairwise kernel (16-16 NN)	≈300	Medium
General symmetric NN	≈5,000	High
Standard Neural ODE	≈10,000	Highest

Table 3: Parameter counts for N(P)X architectures vs. standard Neural ODE.

The dynamics $\dot{X} = NX$ have a clear physical interpretation: node i's velocity is $\dot{X}_i = \sum_j N_{ij} X_j$, a weighted combination of all positions where $N_{ij} > 0$ indicates attraction and $N_{ij} < 0$ indicates repulsion. When the true dynamics have this form, the polynomial parameterization can recover the exact coefficients with orders of magnitude fewer parameters than a generic neural network.

2.5 Realizable Dynamics and Model Diagnostics

Beyond gauge freedom, RDPG dynamics face a fundamental geometric constraint: the probability matrix $P = XX^\top$ lives on a low-dimensional manifold, so most symmetric perturbations $\dot{P}$ are not achievable.

Proposition 3 (Tangent Space Constraint). Let $V \in \mathbb{R}^{n\times d}$ be an orthonormal basis for $\mathrm{col}(P) \subset \mathbb{R}^n$, and $V_\perp \in \mathbb{R}^{n\times(n-d)}$ span its orthogonal complement. Any realizable $\dot{P}$ satisfies:

$$V_\perp^\top \dot{P}\, V_\perp = 0$$

The realizable tangent space has dimension $nd - \frac{d(d-1)}{2}$.

Proof. Any realizable $\dot{P} = FX^\top + XF^\top$ for some $F \in \mathbb{R}^{n\times d}$. Since $\mathrm{col}(X) = \mathrm{col}(V)$, we have $X = VR$ for invertible $R \in \mathbb{R}^{d\times d}$. Then:

$$V_\perp^\top \dot{P}\, V_\perp = V_\perp^\top F R^\top V^\top V_\perp + V_\perp^\top V R F^\top V_\perp = 0 + 0 = 0$$

using $V^\top V_\perp = 0$. Conversely, any symmetric $\dot{P}$ with $V_\perp^\top \dot{P} V_\perp = 0$ can be written in this form. □

Interpretation of blocks. Any symmetric $n \times n$ matrix M decomposes as:

$$M = \underbrace{VAV^\top}_{\text{range-range}} + \underbrace{VBV_\perp^\top + V_\perp B^\top V^\top}_{\text{range-null cross}} + \underbrace{V_\perp C V_\perp^\top}_{\text{null-null}}$$

For realizable $\dot{P}$: the A and B blocks can be arbitrary, but $C = 0$ always.

The null-null block represents "structure in the orthogonal complement"—dynamics that would increase the rank of P. If we fit an RDPG model and find systematic residuals with $C \neq 0$, this suggests the latent dimension d is too small or the RDPG model is inappropriate.

Corollary 3 (Dimension Count).

$$\dim(T_P \mathcal{M}_d) = nd - \frac{d(d-1)}{2}$$

This equals the dimension of X-space (nd) minus the gauge freedom ($\frac{d(d-1)}{2}$)—exactly the observable degrees of freedom.

Example. For $n = 10$ nodes and $d = 2$ dimensions: symmetric matrices have 55 degrees of freedom, but only 19 directions are realizable. The remaining 36 directions would require increasing the rank of P—that is, increasing the latent dimension d.

Model diagnostic. This constraint provides a principled diagnostic for model adequacy. If observed dynamics have structure in the "null-null" block $V_\perp^\top \dot{P} V_\perp$, this indicates one of two possibilities:

1. **Model misspecification**: The true dynamics do not preserve low-rank structure, so RDPG embedding is inappropriate.
2. **Dimensional emergence**: The latent dimension d is increasing over time—new factors are emerging in the network structure.

In practice, the constraint is automatically satisfied by dynamics of the form $\dot{X} = N(P)X$, since $\dot{P} = NP + PN$ has the required tangent structure by construction. Violations in fitted residuals suggest the chosen d may be too small.

2.6 Probability Constraints

For edge probabilities to be valid, we require $P_{ij} \in [0, 1]$ for all i, j. A sufficient condition is that all node positions lie in the positive orthant of the unit ball:

$$B_+^d = \{x \in \mathbb{R}^d : x \geq 0, \|x\| \leq 1\}$$

If $X_i \in B_+^d$ for all i, then $P_{ij} = X_i \cdot X_j \in [0, 1]$ by non-negativity of coordinates and Cauchy-Schwarz.

However, $\left(B_+^d\right)^n$ is *not* a fundamental domain: one cannot always rotate n vectors simultaneously into the positive orthant. For example, two orthogonal unit vectors in $\mathbb{R}^2$ cannot both have non-negative coordinates after any rotation.

We handle constraints via a barrier loss:

$$\mathcal{L}_{\text{prob}} = \gamma \sum_{i,j} \left[\max\left(0, -P_{ij}\right)^2 + \max\left(0, P_{ij} - 1\right)^2\right]$$

This encourages learned dynamics to remain in the valid region without breaking the ODE structure. Unlike projection-based approaches that clamp values, the barrier loss maintains differentiability throughout training.

Practical note. While the B_+^d constraint provides a convenient sufficient condition for mathematical analysis, our experiments show that explicit projection onto B_+^d is neither necessary nor beneficial for numerical learning. Euclidean ODE solvers with barrier losses maintain valid probabilities while preserving the natural geometry of the embedding space. Projecting onto B_+^d can distort the geometry and introduce artifacts that complicate learning. The B_+^d framework remains valuable for theoretical analysis of constraint satisfaction, but practitioners should prefer unconstrained optimization with soft penalties.

2.7 Symbolic Regression

The trained Neural ODE provides accurate predictions but remains a black box. To extract interpretable dynamics, we apply symbolic regression to find closed-form expressions approximating f_θ.

Given the learned vector field $f_{\theta(\boldsymbol{u})}$, we seek symbolic expressions $g(\boldsymbol{u})$ from a grammar of operations (polynomials, trigonometric functions, etc.) that minimize:

$$\min_g \int \|f_{\theta(\boldsymbol{u})} - g(\boldsymbol{u})\|^2 \,\mathrm{d}\boldsymbol{u} + \text{complexity}(g)$$

The complexity penalty encourages parsimonious expressions. We use genetic programming [11] to search the space of symbolic expressions.

Gauge dependence of symbolic equations. A critical caveat: recovered equations are *gauge-dependent*. The symbolic form depends on the coordinate system fixed by SVD and Procrustes alignment. In a rotated basis, $\dot{X}_1 = \omega X_2$ might appear as $\dot{Y}_1 = aY_1 + bY_2$—different-looking equations producing identical $P(t)$.

What *is* gauge-invariant:

- Eigenvalues of the linearization (frequencies, decay rates)
- Equilibrium structure (existence, stability type)
- Qualitative behavior (oscillatory, stable, chaotic)

For truly coordinate-free equations, one could regress $\dot{P}_{ij}$ directly as functions of P, though at the cost of higher dimensionality.

3 Data: Synthetic Temporal Networks

We evaluate our framework on three synthetic temporal networks with known generating processes. This allows us to assess whether extracted equations match ground truth. Crucially, all three systems exhibit *observable* dynamics in the sense of Section 2.2 —the latent position changes produce actual changes in P.

3.1 Single Community Oscillation

A network of $n = 5$ nodes where connection probabilities oscillate sinusoidally. All nodes belong to a single community whose internal connectivity varies periodically. The ground-truth dynamics follow:

$$\frac{dL_{i,1}}{dt} = \omega L_{i,2}, \quad \frac{dL_{i,2}}{dt} = -\omega L_{i,1}$$

producing circular trajectories in embedding space. This is observable because nodes circulate around a nonzero centroid (Proposition 1), not the origin.

3.2 Two Communities Merging

A network of $n = 11$ nodes initially partitioned into two separate communities. Over time, the communities gradually merge into a single cohesive group. This models scenarios like organizational mergers or ecosystem succession. The dynamics involve attraction between nodes (Table 1), which is observable as it changes pairwise dot products.

3.3 Long-Tailed Degree Distribution

A network of $n = 36$ nodes with a power-law degree distribution. This tests whether our method handles networks with heterogeneous node connectivity, which are common in real-world systems due to preferential attachment.

4 Results

4.1 Training Performance

Table 4 summarizes the training results across all three systems.

Dataset	n	d	**Final MSE**
Single community oscillation	5	2	0.114
Two communities merging	11	2	1.169
Long-tailed distribution	36	2	0.159

Table 4: Training results for three synthetic temporal networks.

The single community and long-tail systems achieve low reconstruction error, while the merging communities system is more challenging. This may reflect the more complex dynamics of community reorganization.

4.2 Embedding Trajectories

Figure below (placeholder) shows example embedding trajectories comparing ground truth (from data) with Neural ODE predictions. The model successfully captures the qualitative dynamics in all cases.

4.3 Symbolic Regression

For the single community oscillation system, symbolic regression recovers equations of the form:

$$\frac{dL_1}{dt} \approx aL_2, \quad \frac{dL_2}{dt} \approx -aL_1$$

matching the ground-truth harmonic oscillator dynamics. This demonstrates that our framework can recover interpretable, mechanistically meaningful equations from network observations alone.

4.4 Gauge-Consistent Architecture Comparison

To test whether the theoretically-motivated $\dot{X} = N(P)X$ form offers practical advantages, we compare three architectures on synthetic pairwise dynamics:

$$\dot{X} = (\alpha I + \beta P)X$$

with $\alpha = -0.02$ (slight contraction) and $\beta = 0.001$ (pairwise attraction). This dynamics has exactly the polynomial $N(P)X$ form with degree 1, providing a fair test where the correct inductive bias should help.

We compare:
1. **Standard Neural ODE**: Generic $f_{\theta(X)}$ with ≈10,000 parameters
2. **Polynomial $N(P)X$**: $N = \alpha_0 I + \alpha_1 P$ with 2 parameters
3. **Kernel $N(P)X$**: $N_{ij} = \kappa(P_{ij})$ with ≈300 parameters

Table 5 summarizes the results.

Architecture	**Parameters**	**MSE**	**Parameter Recovery**
Standard Neural ODE	≈10,000	—	N/A
Polynomial $N(P)X$	2	—	$\hat{\alpha}_0 \approx ?$, $\hat{\alpha}_1 \approx ?$
Kernel $N(P)X$	≈300	—	N/A

Table 5: Architecture comparison on pairwise dynamics ($n = 30$, $d = 2$).

The polynomial architecture offers a dramatic reduction in parameters (5000× fewer than standard NN) while potentially recovering the true dynamical coefficients. When the inductive bias matches the true dynamics, parsimony does not sacrifice accuracy.

5 Discussion

We presented a framework for learning interpretable dynamics of temporal networks. By combining RDPG embedding, Neural ODEs, and symbolic regression, we bridge the gap between black-box prediction and mechanistic understanding.

Theoretical foundations. The gauge-theoretic analysis (Section 2.2) provides principled answers to fundamental questions: *What can we learn?* All dynamics except uniform rotation around the origin (Theorem 1). *What architecture respects the structure?* The form $\dot{X} = N(P)X$ with symmetric N (Theorem 6). These results inform both architecture design and interpretation of learned models.

Practical obstructions. Beyond the theoretical gauge freedom, practical challenges include: (i) estimation error in $\widehat{X}$ from SVD (≈35% position error, though $\hat{P}$ has only ≈5% error); (ii) Procrustes alignment artifacts that can introduce spurious motion or remove real motion resembling global rotation; (iii) discrete, noisy observations rather than continuous $P(t)$. These factors may explain why some dynamics (e.g., circulation) are harder to learn than others (e.g., attraction/ repulsion).

Evaluation in $\boldsymbol{P}$-space. The gauge freedom implies that X-based metrics (e.g., position RMSE) are coordinate-dependent and potentially misleading. The natural evaluation metric is $P(t) = X(t)X(t)^\top$, which is gauge-invariant. Comparing predicted $\hat{P}(t)$ to true $P(t)$ directly tests whether the learned dynamics capture observable network structure, independent of coordinate conventions. Our experiments confirm that models can achieve low P-error even when X-trajectories differ substantially due to gauge ambiguity.

Parsimonious architectures. The $\dot{X} = N(P)X$ architecture with symmetric N offers two key advantages over generic neural networks: (1) automatic gauge consistency—symmetric N cannot produce invisible dynamics (Theorem 6), and (2) dramatic parameter reduction—polynomial N achieves comparable accuracy with 10^3–10^4 fewer parameters. When the true dynamics have this form, the polynomial parameterization can recover exact coefficients, providing interpretability that symbolic regression cannot match for black-box neural networks.

Model diagnostics. The tangent space constraint (Proposition 3) provides a diagnostic for model adequacy. If residuals $\dot{\hat{P}} - \dot{P}_{\text{pred}}$ have systematic structure in the null-null block $V_\perp^\top(\cdot)V_\perp$, this suggests either that the RDPG model is inappropriate or that the latent dimension d should be increased. This connects temporal network modeling to static dimension selection methods, potentially enabling dynamic diagnostics for detecting emerging community structure.

Limitations. The long-tailed network shows higher reconstruction error, suggesting challenges with highly heterogeneous degree distributions. The polynomial $N(P)X$ form, while parsimonious, may be too restrictive for dynamics that do not factor through P. For such cases, the kernel or general symmetric architectures provide a middle ground. Additionally, all methods rely on accurate RDPG embedding, which introduces estimation error (≈35% in positions, though only ≈ 5% in probabilities).

Extensions. The UDE framework (Section 2.4) enables incorporating domain knowledge. For ecological networks, one might encode known trophic interactions in N_{known} while learning corrections. For social networks, community structure could inform block-diagonal parameterizations. The theory extends to directed graphs (Section 10), where $P = LR^{\top}$ with separate dynamics for source and target embeddings, and to oscillatory dynamics (Section 9), which symmetric N can produce through nonlinear coupling despite having real eigenvalues in the linear case.

6 Conclusion

We introduced a framework that transforms the problem of temporal network modeling from discrete event prediction to continuous dynamical systems analysis. The gauge-theoretic analysis reveals that RDPG embeddings have inherent rotational ambiguity, but we identify a broad class of observable dynamics and derive architectures that are gauge-consistent by construction.

The parsimonious $\dot{X} = N(P)X$ form with symmetric N achieves two goals simultaneously: it eliminates ambiguity about what the model can learn, and it reduces parameters by orders of magnitude while maintaining or improving accuracy. When the true dynamics have this form, the polynomial parameterization can recover exact coefficients—a level of interpretability that post-hoc symbolic regression cannot match.

Our approach produces interpretable differential equations governing network evolution, enabling both prediction and mechanistic insight. The open-source implementation facilitates application to new domains.

7 Acknowledgments

8 Data and Code Availability

The `RDPGDynamics.jl` package and all data are available at [repository URL]. Experiments can be reproduced with: `julia --project scripts/reproduce_paper.jl`

Bibliography

[1] P. Holme and J. Saramäki, "Temporal networks," *Physics Reports*, vol. 519, no. 3, pp. 97–125, 2012, doi: https://doi.org/10.1016/j.physrep.2012.03.001.

[2] M. Lucas, A. Morris, A. Townsend-Teague, L. Tichit, B. Habermann, and A. Barrat, "Inferring cell cycle phases from a partially temporal network of protein interactions," *Cell Reports Methods*, vol. 3, no. 2, 2023.

[3] S. Hanneke, W. Fu, and E. P. Xing, "Discrete temporal models of social networks," *Electronic Journal of Statistics*, vol. 4, no. none, pp. 585–605, 2010, doi: 10.1214/09-EJS548.

[4] T. Poisot, D. B. Stouffer, and D. Gravel, "Beyond species: why ecological interaction networks vary through space and time," *Oikos*, vol. 124, no. 3, pp. 243–251, 2015.

[5] M. A. Porter and J. P. Gleeson, "Dynamical systems on networks," *Frontiers in Applied Dynamical Systems: Reviews and Tutorials*, vol. 4. Springer, 2016.

[6] P. Holme, "Modern temporal network theory: a colloquium," *The European Physical Journal B*, vol. 88, no. 9, p. 234, 2015.

[7] S. M. Kazemi *et al.*, “Representation learning for dynamic graphs: A survey,” *Journal of Machine Learning Research*, vol. 21, no. 70, pp. 1–73, 2020.

[8] A. Athreya *et al.*, “Statistical inference on random dot product graphs: a survey,” *The Journal of Machine Learning Research*, vol. 18, no. 1, pp. 8393–8484, 2018.

[9] R. T. Chen, Y. Rubanova, J. Bettencourt, and D. K. Duvenaud, “Neural ordinary differential equations,” in *Advances in neural information processing systems*, 2018.

[10] C. Rackauckas *et al.*, “Universal Differential Equations for Scientific Machine Learning,” *CoRR*, 2020, [Online]. Available: https://arxiv.org/abs/2001.04385

[11] M. Schmidt and H. Lipson, “Distilling free-form natural laws from experimental data,” *science*, vol. 324, no. 5923, pp. 81–85, 2009.

9 Oscillatory Dynamics with Symmetric N

A common concern: can symmetric $N(P)$ produce oscillations? For a *linear* system $\dot{X} = NX$ with constant symmetric N, eigenvalues are real, so solutions are sums of exponentials—no oscillations.

However, our system $\dot{X} = N(P)X = N(XX^\top)X$ is *nonlinear* because N depends on X through P.

9.1 Linearization Around Equilibrium

At equilibrium X^* with $N(P^*)X^* = 0$, the linearization is:

$$\delta\dot{X} = N(P^*)\delta X + \left[\frac{\partial N}{\partial P}\,|_{P^*} \cdot \left(\delta X \cdot X^{*^\top} + X^* \cdot \delta X^\top\right)\right]X^*$$

The Jacobian (as a linear operator on $\delta X \in \mathbb{R}^{n\times d}$) is *not* simply $N(P^*)$. The second term involves derivatives of N and creates coupling that can produce complex eigenvalues.

9.2 Mechanisms for Oscillation

1. Hopf bifurcation: As parameters vary, eigenvalues of the Jacobian can cross the imaginary axis, creating limit cycles.

2. Amplitude-phase coupling (for $d = 2$): Write $X_i = r_i(\cos\theta_i, \sin\theta_i)$. Then $P_{ij} = r_i r_j \cos(\theta_i - \theta_j)$. Phase differences affect probabilities, which affect phase dynamics—a feedback loop enabling oscillation.

3. Multi-scale interaction:

$$N(P) = \alpha_1 P - \alpha_2 P^2$$

Local attraction ($\alpha_1 P$) competes with nonlocal repulsion ($-\alpha_2 P^2$), potentially creating oscillatory approach to equilibrium.

9.3 What Symmetric N Cannot Do

- Rotation around origin in latent space (this is gauge/invisible anyway)
- Oscillations in the linear approximation with *constant* N

9.4 What Symmetric N Can Do

- Damped oscillations approaching equilibrium (via nonlinear Jacobian)
- Limit cycles via Hopf bifurcation
- Quasi-periodic motion in higher dimensions

10 Extension to Directed Graphs

For directed graphs, the probability matrix factors as $P = LR^\top$ where $L, R \in \mathbb{R}^{n\times d}$ are **left** (source) and **right** (target) embeddings.

10.1 Gauge Group

The gauge transformation is $(L, R) \mapsto (LQ, RQ)$ for $Q \in O(d)$:

$$(LQ)(RQ)^\top = LQQ^\top R^\top = LR^\top = P$$

Crucially, both embeddings rotate by the *same* Q.

10.2 Gauge-Invariant Quantities

Under $(L, R) \mapsto (LQ, RQ)$:

- $P = LR^\top$ — invariant
- $G_L = LL^\top$ — invariant
- $G_R = RR^\top$ — invariant
- $L^\top L$, $R^\top R$, $L^\top R$ — transform by conjugation

The gauge-invariant data is (P, G_L, G_R)—three $n \times n$ matrices.

10.3 Equivariant Dynamics

Any $O(d)$-equivariant vector field has the form:

$$\dot{L} = N_{L(P,G_L,G_R)} \cdot L, \qquad \dot{R} = N_{R(P,G_L,G_R)} \cdot R$$

To eliminate gauge, require $N_L = N_L^\top$ and $N_R = N_R^\top$ individually.

10.4 Induced Dynamics on P

$$\dot{P} = N_L P + P N_R^\top$$

This is a Sylvester equation, enabling:

- Asymmetric growth patterns
- Directed community formation
- Source-target differentiation

Special case: If $N_L = N_R = N$ with N symmetric, the directed dynamics reduce to the undirected form $\dot{P} = NP + PN$.

11 Parsimonious UDE Parameterizations

This appendix catalogs parameterization choices for $N(P)$ in the UDE framework $\dot{X} = N(P)X$.

11.1 Taxonomy by Homogeneity

Type	**Description**	**Parameters**
Homogeneous	All node pairs follow same rule	$O(1)$ functions
Type-based	Nodes grouped into K types	$O(K^2)$ functions
Node-specific	Each node has own parameters	$O(n)$ scalars
Fully heterogeneous	Each pair has own parameter	$O(n^2)$—avoid!

Table 6: Parameterization complexity by homogeneity assumption.

11.2 Homogeneous Parameterizations

Polynomial in P:

$$N(P) = \alpha_0 I + \alpha_1 P + \alpha_2 P^2 + \dots + \alpha_k P^k$$

Parameters: $k + 1$ scalars. Interpretation: $\alpha_0 I$ is self-dynamics, $\alpha_1 P$ is direct neighbor influence, $\alpha_2 P^2$ is two-hop influence.

Pairwise kernel:

$$N_{ij}(P) = \kappa\big(P_{ij}\big) \quad \text{for} \;\; i \neq j, \qquad N_{ii}(P) = h(P_{ii})$$

Symmetry is automatic since $P_{ij} = P_{ji}$. The function κ can be a small neural network or parametric (e.g., $\kappa(p) = a + bp + cp^2$).

Attraction-repulsion (Lennard-Jones inspired):

$$\kappa(p) = \frac{a}{p + \varepsilon} - \frac{b}{(p + \varepsilon)^2}$$

Parameters: (a, b, ε). Equilibrium occurs where attraction balances repulsion.

Laplacian-based:

$$N(P) = \alpha\big(D^{-1/2} P D^{-1/2} - I\big)$$

where $D = \operatorname{diag}(P\mathbf{1})$ is the degree matrix. This encodes normalized diffusion.

11.3 Type-Based Parameterizations

When nodes belong to types $\tau : \{1, ..., n\} \to \{1, ..., K\}$, interactions can depend on type pairs.

Block kernel:

$$N_{ij} = \kappa_{\tau(i),\tau(j)}\big(P_{ij}\big)$$

Symmetry requires $\kappa_{ab} = \kappa_{ba}$. Parameters: $\frac{K(K+1)}{2}$ functions.

Stochastic Block Model prior:

$$N_{ij} = \alpha_{\tau(i),\tau(j)} + \beta \cdot P_{ij}$$

Parameters: $\frac{K(K+1)}{2}$ scalars α_{ab} plus one shared β.

Interpretation: Base rate depends on community pair, plus universal connection-strength effect.

11.4 Node-Specific Parameterizations

Each node has individual parameters, but interactions follow shared rules.

Diagonal + shared off-diagonal:

$$N_{ij} = \begin{cases} h_i & i = j \\ \kappa\big(P_{ij}\big) i \neq j \end{cases}$$

Parameters: n scalars h_i (node-specific self-rates) plus 1 function κ (shared interaction).

Node features determine rate:

$$h_i = g(\varphi_i), \quad \varphi_i = \left(P_{ii}, \sum_j P_{ij}, \max_j P_{ij}, ...\right)$$

where φ_i are node-level features extracted from P and g is a shared function (can be a small NN).

11.5 Message-Passing Formulation

An equivalent view writes dynamics as node-level updates:

$$\dot{X}_i = a(P_{ii})X_i + \sum_{j \neq i} m(P_{ij})(X_j - X_i)$$

where a is the intrinsic rate and m is the message function.

The equivalent symmetric N is:

$$N_{ij} = \begin{cases} a(P_{ii}) - \sum_{k \neq i} m(P_{ik}) i = j \\ m(P_{ij}) & i \neq j \end{cases}$$

11.6 Low-Rank Parameterizations

Rank-r symmetric:

$$N = \sum_{k=1}^{r} \alpha_k u_k u_k^\top = U \text{ diag}(\alpha) U^\top$$

Parameters: $nr + r$ (with orthogonality constraints on U).

Data-derived basis:

$$N = \sum_{k=1}^{r} \alpha_k v_k v_k^\top$$

where v_k are the top eigenvectors of P itself. Parameters: r scalars only.

11.7 Encoding Qualitative Priors

Prior	**Parameterization**
Stability (nodes don't explode)	$N = -\exp(M)$ where $M = M^\top$
Conservation (tr(P) constant)	Project N to $\text{tr}(NP) = 0$
Known equilibrium P^*	$N(P) = (P - P^*)M(P)$
Sparsity preservation	$N_{ij} = P_{ij} \cdot \kappa(P_{ij})$

Table 7: Parameterizations encoding specific priors.

11.8 Geometric Boundary Constraints

There are two independent sources of constraints on $\dot{P}$: algebraic (rank preservation, automatically satisfied by $\dot{X} = N(P)X$) and geometric (probability bounds).

At lower boundary ($P_{ij} = 0$): Require $\dot{P}_{ij} \geq 0$.

From $\dot{P} = NP + PN$:

$$\dot{P}_{ij} = \sum_k N_{ik} P_{kj} + \sum_k P_{ik} N_{kj}$$

Caution: This is NOT simply a Metzler condition on N. For the linear system $\dot{y} = Ay$, Metzler A (non-negative off-diagonal) preserves the positive orthant. But $\dot{P} = NP + PN$ is a Lyapunov equation—the condition involves the entire structure of P, not just local properties of N.

At upper boundary ($\boldsymbol{P_{ij} = 1}$): Require $\dot{P}_{ij} \leq 0$.

Since $P_{ij} = X_i \cdot X_j \leq \|X_i\| \, \|X_j\|$, we have $P_{ij} = 1$ only if $X_i = X_j$ with $\|X_i\| = 1$.

Practical enforcement: Rather than modifying N (which breaks the symmetric structure) or projecting onto B^d_+ (which distorts the geometry), use a barrier in the loss function:

$$\mathcal{L}_{\text{barrier}} = \gamma \sum_{i,j} \Big[\max\big(0, -P_{ij}\big)^2 + \max\big(0, P_{ij} - 1\big)^2\Big]$$

This encourages learned dynamics to stay in the valid region while preserving the natural geometry of the embedding space. Our experiments show that projection-based constraint enforcement (e.g., onto B^d_+) is unnecessary and can actually impede learning.

Summary: In the interior of the valid configuration space, only the algebraic constraint matters—and it's automatic. Geometric constraints only matter at boundaries. Soft penalties (barrier loss) outperform hard constraints (projection) in practice.