Experimental Design Plan

For Project: *Auto-Associative Neural Networks for Protein Interaction Dynamics*
Prepared by: Sreevatsa Nukala

1 Objectives and Success Criteria

Reproduction Goals

- R1: Sigmoid AANN with known signed connectivity fits trajectories accurately.
- R2: Linear AANN underfits; hybrid (linear + sigmoid) performs comparably to sigmoid.
- R3: Recurrent AANN improves temporal predictions; relies on past state.
- R4: "Self-learning" (no teacher forcing) drifts and accumulates error.

Extension Goals

- X1: Analyze stability via Jacobians, fixed points, and eigenvalue spectra.
- X2: Implement and evaluate Contrastive Hebbian Learning (CHL) vs gradient descent.
- X3: Investigate unknown connectivity + sparsity (L1 regularization, identifiability).
- X4: Quantify memory contribution from recurrent terms.
- X5 (optional): Robustness and Hopfield-style energy basins.

Success = Complete all R1–R3 + any two of X1–X4 with figures and concise theoretical interpretation.

2 Data Generation

2.1 Signed Wiring

- Signed adjacency $S \in \{-1, 0, +1\}^{12\times 12}$ (from supplement).
- Sample magnitudes $|W|_{ij} \sim \text{LogNormal}(\mu_w = -1.2, \sigma_w = 0.6)$ for $S_{ij} \neq 0$.
- $W_{\text{eff}} = S \odot |W|$, rescaled so $\rho(\alpha W_{\text{eff}}) \leq 1$.

2.2 Simulator A: Discrete-Time Sigmoid Network

$$x_{t+1} = \sigma\big(\alpha W_{\text{eff}} x_t + b + \eta_t\big), \quad \eta_t \sim \mathcal{N}(0, \sigma_\eta^2 I)$$

- Parameters: $\alpha = 0.9$, $\sigma_\eta = 0.02$, $x_0 \sim \text{Beta}(2, 2)^{12}$.
- Bias: $b = \sigma^{-1}(\mu) - \alpha W_{\text{eff}}\mu$, with $\mu \in [0.3, 0.7]$.

2.3 Simulator B: Continuous-Time ODE (Optional)

$$\dot{x} = -\tfrac{1}{\tau}x + \sigma(\alpha W_{\text{eff}} x + b) + \xi(t), \quad \tau = 1,\ \alpha = 1$$

2.4 Simulator C: Binary Network (Optional)

$$x_{t+1}^{(B)} = \not{\mathbb{1}}\Big[W^{(B)} x_t^{(B)} - \theta > 0\Big]$$

with thresholds θ_i matched to node in-degree.

2.5 Data Structure and Splits

- Generate $T = 400$ steps $\times$ 10 random seeds.
- Split: 70% train / 15% val / 15% test (time-ordered).
- Produce:
  - $\mathcal{D}^{(1)} = \{(x_t, x_{t+1})\}$ for non-recurrent.
  - $\mathcal{D}^{(2)} = \{([x_t; x_{t-1}], x_{t+1})\}$ for recurrent.
  - $\mathcal{D}^{(B)} = \{(x_t, y^{(B)}_{t+1})\}$ for binary.

3 Models

- **Non-recurrent AANN:** $y = \phi(Wx + b)$, with mask $M = \mathbb{1}[S \neq 0]$.
- **Hybrid AANN:** per-node activations (sigmoid/linear) chosen by single-protein fits.
- **Recurrent AANN:** $x_{t+1} = \phi(Wx_t + Ux_{t-1} + b)$, U diagonal or learned.

4 Training Procedures

4.1 Manual NumPy Training (Non-recurrent)

- Loss: $\mathrm{MSE}(y, x_{t+1})$.
- Gradients: $dz = 2(y - x_{t+1})/d \odot \phi'(z)$, $dW = dz x_t^\top \odot M$, $db = dz$.
- Optimizer: SGD + momentum ($\mu = 0.9$, $\eta \in \{10^{-2}, 5\times10^{-3}, 10^{-3}\}$).

4.2 Short Unrolled Recurrent Training

- Unroll $T = 5$–10 steps, accumulate gradients, clip at 1.0.
- Optional PyTorch autograd if NumPy BPTT unstable.

4.3 Contrastive Hebbian Learning (CHL)

- Free phase: converge to r^F under input.
- Clamped phase: converge to r^C under target.
- Update: $\Delta W = \eta(r^C r^{C\top} - r^F r^{F\top}) \odot M$, $\Delta b = \eta(r^C - r^F)$.

4.4 Unknown Connectivity + L1 (ISTA)

$$W \leftarrow \mathrm{soft}(W - \eta \nabla \mathrm{MSE}, \eta\lambda), \quad \lambda \in \{10^{-3}, 3\times10^{-3}, 10^{-2}\}$$

5 Experiments

R1. Known-Graph Sigmoid vs Linear

Compare MSE, correlations, overlays (3–4 proteins). *Expected: Sigmoid $\ll$ Linear; reproduces paper's main result.*

R2. Hybrid Model

Per-node activation assignment from single-protein fits. *Expected: Hybrid $\approx$ Sigmoid accuracy.*

R3. Recurrent Model

Unroll 5–10 steps; ablate U. *Expected: Recurrent outperforms non-recurrent; ablation increases error.*

R4. Self-Learning

Roll model on own outputs without teacher forcing. *Expected: Drift and instability.*

X1. Stability and Jacobian Analysis

Compute $J = \text{diag}(\phi'(Wx^* + b))W$ at fixed point x^*. Analyze eigenvalues and stability fraction ($|\lambda| < 1$).

X2. CHL vs SGD

Compare convergence speed and final MSE. *Expected: CHL slower, comparable final accuracy.*

X3. Unknown Connectivity + Sparsity

Train with full W and L1 penalty. Evaluate edge sign accuracy, precision/recall, AUROC.

X4. Memory Quantification

Define Memory Index = $(\text{MSE}_{\text{nonrec}} - \text{MSE}_{\text{rec}})/\text{MSE}_{\text{nonrec}}$. Compute sensitivity via finite differences on x_t vs x_{t-1}.

X5. Robustness (Optional)

Inject weight/state noise; compute stability radius and visualize energy-like basins.

6 Hyperparameters

Parameter	**Values / Defaults**
Simulator gain α	0.8, **0.9**, 1.0
Process noise σ_η	0.0, **0.02**, 0.04
Bias mean μ	0.4, **0.5**, 0.6
LR	1e–2, 5e–3, 1e–3
Epochs	max 300, patience 20
Unroll steps T	5, **8**, 10
CHL step size η	1e–3, **3e–3**, 1e–2
Seeds	10

7 Metrics and Evaluation

- Primary: Test MSE, per-node correlation.
- Edge recovery: Precision, recall, AUROC vs ground truth S.
- Stability: $\%|\lambda| < 1$, spectral radius of J.
- Statistical tests: paired t-test (sigmoid vs linear MSE).

8 Implementation Layout

```
src/
  data.py           # synthetic simulator
  numpy_core.py     # forward, grads, CHL, ISTA, Jacobians
  models.py         # AANN classes
  analysis.py       # stability, eigs, metrics
notebooks/
  01_repro_linear_vs_sigmoid.ipynb
  02_hybrid_and_single_protein.ipynb
  03_recurrent_and_self_learning.ipynb
  04_extensions_jacobian_chl_unknown.ipynb
```

9 Timeline

- **Day 1–2:** Data + R1 (Sigmoid vs Linear)
- **Day 3:** Hybrid + single-protein fits
- **Day 4:** Recurrent and memory analysis
- **Day 5:** Self-learning test + stability
- **Day 6:** CHL experiments
- **Day 7:** Unknown connectivity + sparsity
- **Day 8:** Memory quantification + robustness

10 Risks and Mitigations

- **Gradient instability:** keep $T \leq 10$; use gradient clipping.
- **CHL non-convergence:** use damping, small η; log ΔE trend.
- **Edge recovery failure:** sweep λ; accept "prediction $\neq$ structure".