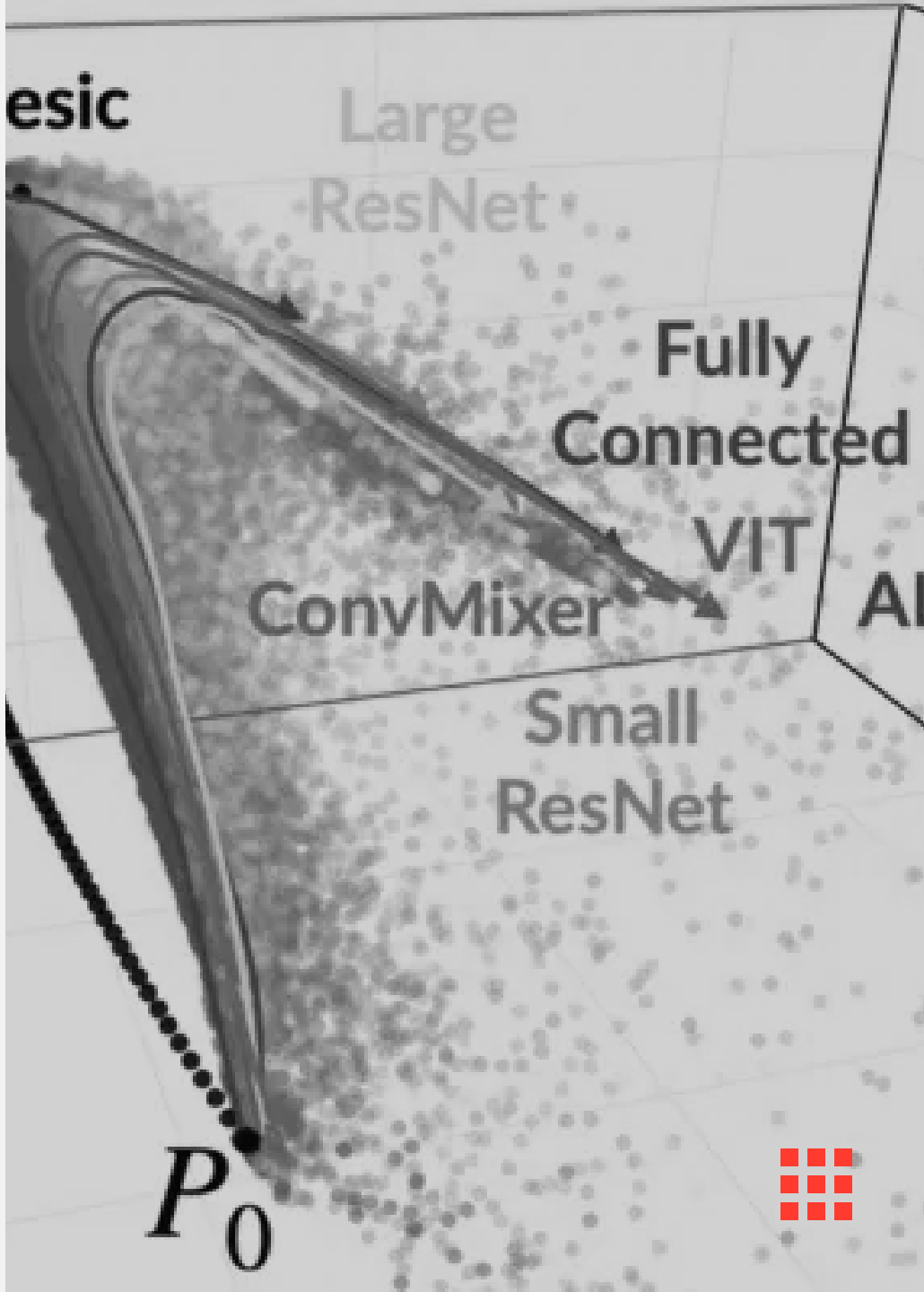
# Trajectory Geometry Determines Algorithmic Structure

> A Paradigm Shift in Neural Network Training

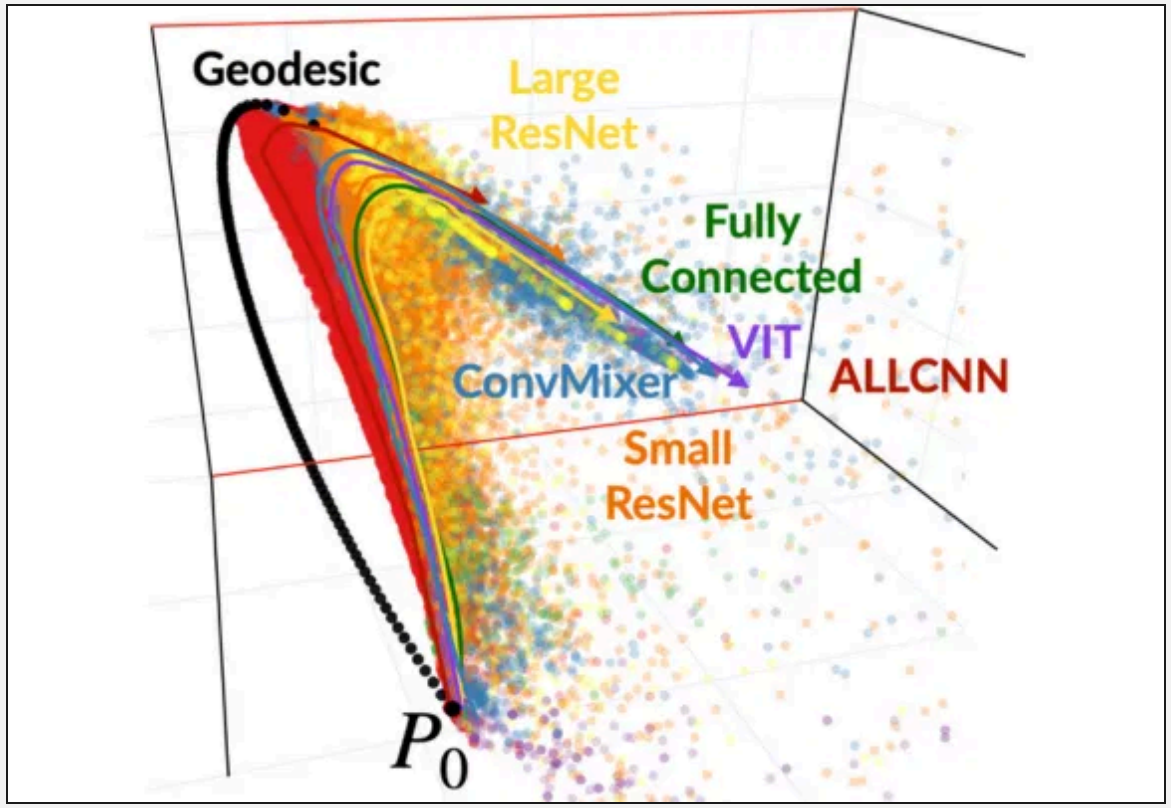
BY GRISUN0



# The Central Paradigm Shift

> **Traditional View:** Algorithmic structure passively emerges from optimization.
>
> **New Paradigm:** Algorithmic structure is **actively constructed** through precise manipulation of gradient dynamics.

| | |
|---|---|
| Controlled Trajectories | **68% Success** |
| Uncontrolled (Local Minima) | **32% Failure** |
| Critical Batch Size | **[24, 128]** |




Visual representation of training trajectories. Identical final loss values can result from different paths, but only specific geometries reach genuine algorithmic solutions.

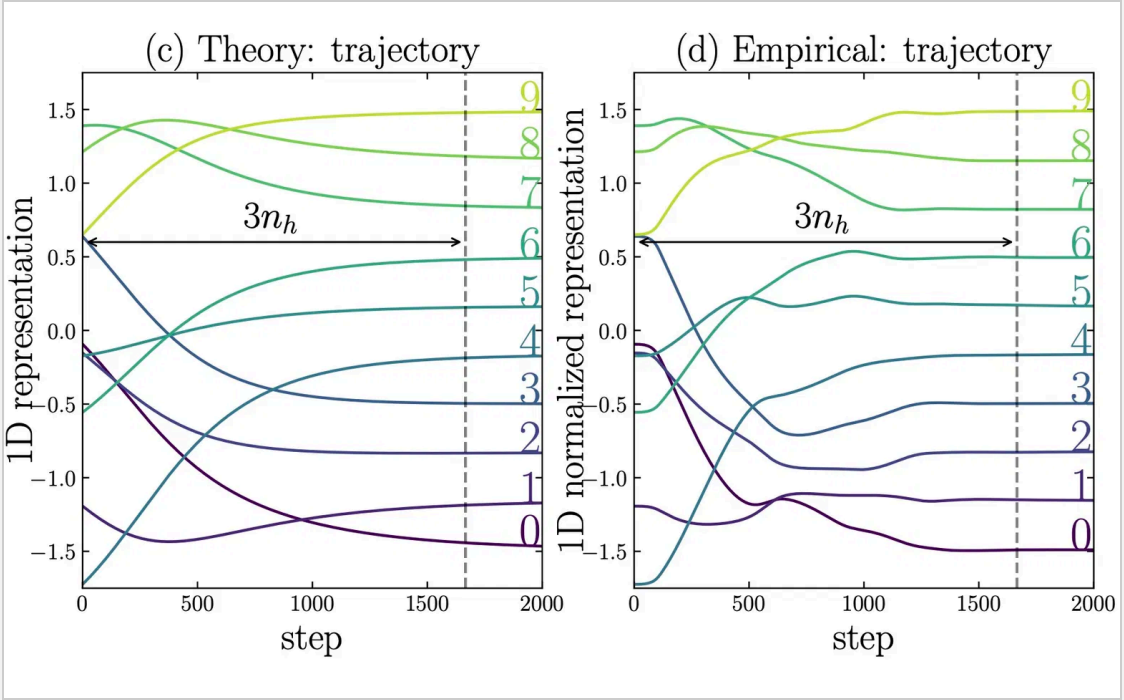
# The Grokking Problem & Verification Challenge

> When a network groks, has it learned the algorithm, or has it found a local minimum that happens to generalize?

## VERIFICATION FRAMEWORK

- **Sparsification:** Prune to exact rank-7 structure (Strassen uses 7 multiplications vs. naive 8).
- **Discretization:** Project weights to integer coefficients in {-1, 0, 1}.
- **Zero-Shot Transfer:** Verify correctness from 2×2 to 64×64 matrices without retraining.

**Only solutions passing all three steps represent genuine algorithmic crystallization.**




Evolution of representations during training. The verification framework distinguishes true structural learning from mere statistical correlation.

# The Two-Phase Protocol for Algorithmic Induction

## 1 Training (Induction)

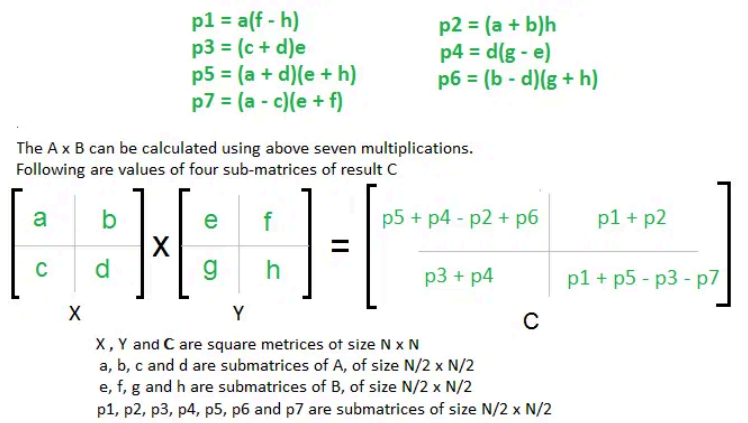
- **Batch Size:** [24, 128] (Critical for trajectory geometry)
- **Optimizer:** AdamW with weight decay ≥ 1e-4
- **Duration:** 1000+ epochs (Until grokking occurs)
- **Goal:** Reach basin of attraction

## 2 Verification (Crystallization)

- **Sparsification:** Prune to exactly 7 active slots
- **Discretization:** Round weights to {-1, 0, 1}
- **Validation:** Verify matrix multiplication correctness

**CRITICAL INSIGHT**

Phase 1 alone achieved 0% success in early experiments. Explicit sparsification (Phase 2) is required to crystallize the approximate solution into the exact algorithm.




| Feature | Type |
|---|---|
| Rank-7 Constraint | **Engineered** |
| Integer Coeffs | **Engineered** |
| Zero-shot Transfer | Emergent |
| Discrete Values | Mixed |

# The Batch Size Investigation

**01** **OBSERVATION**

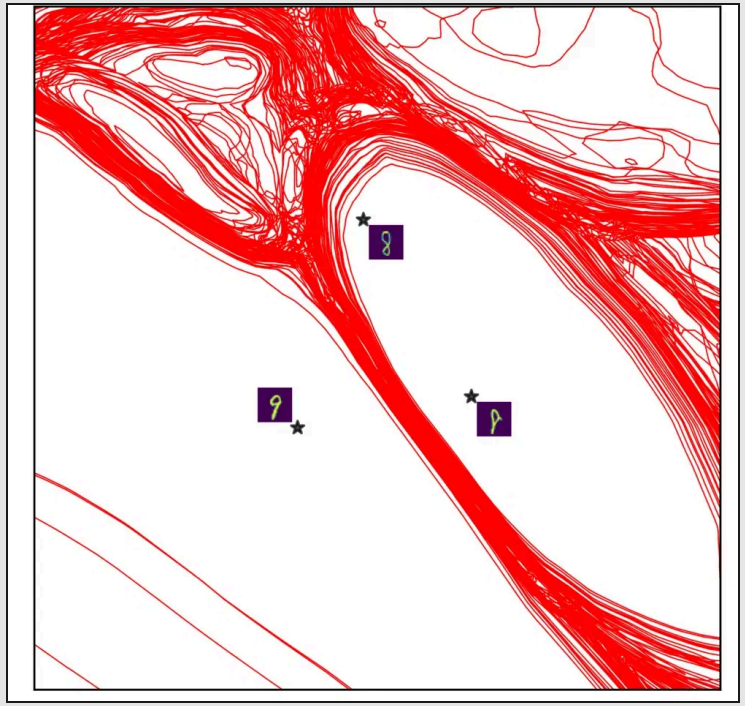
Batch sizes in **[24, 128]** achieve 68% crystallization success. Other values largely fail.

**02** **HYPOTHESIS (REJECTED)**

Hardware cache effects?
**Evidence Against:** Memory analysis shows even B=1024 fits comfortably in L3 cache (321 KB vs ≥1MB).

**03** **TRUE MECHANISM**

**Gradient Covariance Geometry.** Optimal batch size stabilizes the condition number, creating trajectories where noise balances exploration and stability.



**Paradigm Demonstration:** Solutions at B=32 and B=512 may have identical loss, but only specific trajectory geometries reach the narrow basin containing the algorithm.

# Extreme Fragility Reveals Narrow Basins

Algorithmic solutions occupy **extremely narrow basins** in weight space.

Weights must converge precisely to integer values. Even microscopic deviations prevent structural crystallization.

### BROADER IMPLICATIONS

> Reproducibility failures may reflect missed basins, not architectural flaws.

> "Trajectory Engineering" is required to steer dynamics into these narrow regions.

> Standard hyperparameter tuning is insufficient without geometric control.

## Noise Stability Test

| NOISE (Σ) | SUCCESS RATE |
|---|---|
| 0.001 | 0% |
| 0.005 | 0% |
| 0.010 | 0% |
| 0.050 | 0% |

Critical Threshold

**$\sigma < 0.001$**

# Statistical Validation & Empirical Robustness

Analysis of 195 training runs confirms that batch size is a dominant control parameter for algorithmic crystallization.

# Zero-Shot Transfer Validates Genuine Algorithmic Learning

| MATRIX SIZE | MAX ERROR | STATUS |
|---|---|---|
| 2 × 2 | 2.38e-07 | **PASS** |
| 4 × 4 | 1.91e-06 | **PASS** |
| 8 × 8 | 6.20e-06 | **PASS** |
| 16 × 16 | 2.15e-05 | **PASS** |
| 32 × 32 | 8.13e-05 | **PASS** |
| 64 × 64 | 2.94e-04 | **PASS** |

## WHY THIS MATTERS

Zero-shot transfer is the **definitive test**. A network that merely memorized patterns or found a convenient local minimum would fail catastrophically when expanded to larger matrices.

## CONTRAST WITH FAILED RUNS

The 32% of runs that fail verification achieve low test loss on 2×2 matrices but **cannot transfer**. They learned generalizing local minima, not the true Strassen algorithm.

# Implications for Deep Learning Reproducibility

Reframing reproducibility as a trajectory engineering problem.

### THE CHALLENGE

Many results are hard to reproduce even with identical hyperparameters.

Standard explanations like implementation details or data variations are often **insufficient** to explain why one run succeeds and another fails.

### THE INSIGHT

**Trajectory Geometry** is the missing factor.

Random initialization or hardware noise leads to different trajectories. If the target solution is in a **narrow basin**, one trajectory may reach it while another settles into a nearby local minimum.

### THE SHIFT

We must move from "Hyperparameter Tuning" to **Trajectory Engineering**.

It is necessary to understand which parameters control geometry and how to **steer trajectories** toward target basins.

# Conclusion

Active Construction Over Passive Emergence

### CORE PARADIGM SHIFT

TRADITIONAL VIEW

Train networks, analyze final solutions, report emergent properties.

↓

NEW VIEW

**Engineer** training trajectories, **verify** structural properties, understand **why** specific conditions succeed.

### KEY CONTRIBUTIONS

02 **Verification Framework:** Explicit criteria distinguish genuine algorithmic learning from generalizing local minima.

04 **Empirical Evidence:** 68% success rate under controlled conditions (195 runs).

06 **Mechanistic Insight:** Batch size effects reflect gradient covariance geometry.

08 **Fragility Documentation:** 0% success with noise $\sigma \geq 0.001$ reveals narrow basins.

010 **Reproducibility:** A new framework based on trajectory engineering.

**Trajectory geometry determines whether networks reach algorithmic solutions. The 32% failure rate reflects the fundamental narrowness of algorithmic basins.**

# References & Reproducibility

## KEY REFERENCES


**Deep Networks Always Grok and Here is Why**

Humayun et al. (2024) • arXiv:2402.15555

**Superposition as Lossy Compression**

Bereska et al. (2024) • arXiv 2024

**Algorithmic Induction via Structural Weight Transfer**

grisun0 (2025) • Zenodo DOI: 10.5281/zenodo.18263654


## REPRODUCTION

github.com/grisuno/strass_strassen

```
git clone https://github.com/grisuno/strass_strassen
cd strass_strassen
pip install -r requirements.txt
python app.py
```

### SUCCESS CRITERIA (PRE-TRAINED CHECKPOINTS)

- ✓ $\delta = 0$ (Weights exactly integers in {-1, 0, 1})
- ✓ Max Error < 1e-5 (Correct multiplication)
- ✓ $S(\theta) = 1$ (Successful crystallization)