The Shape of Learning in Repeated Games

Information-Geometric Analysis via InPCA

Nora Han

Supervised by Prof. Aaron Roth

Learning in Games
CIS 6200 Advanced Topics in Machine Learning

Dec 4, 2025

Penn
University *of* Pennsylvania

Outline

Intrduction

- 1.1 Problem Setting
- 1.2 Why Learning Trajectories Matter

1.1 Problem Setting

- Repeated-play games: agents update strategies over time.
- Standard analysis tracks regret and convergence, not *how* strategies move.
- Learning algorithms (MWU, OGD, etc.) produce complex trajectories in probability space.
- Existing projection (naive PCA) distort or hide this structure.

1.2 Why Learning Trajectories Matter

- Last-iterate may cycle even when time-average converges.
- Trajectory geometry reveals exploitability, stability, and dynamic modes.
- Different algorithms trace distinct shapes—this is not captured by regret alone.
- Understanding geometry => understanding algorithmic behavior.

Core Question

What hidden geometric structure underlies learning in repeated games?

Background and Prior Work

- 2.1 Classical Results on Convergence in Repeated Games
- 2.2 No-Regret Learning Rules and Known Behaviors
- 2.3 Limitations of Existing Visualization and Analysis Approaches

2.1 Classical Results on Convergence in Repeated Games

Repeated zero-sum game setup

- Two players (row, column), finite action sets A_1, A_2 with $|A_1| = m$, $|A_2| = n$.
- Mixed strategies $p_t \in \Delta_m$, $q_t \in \Delta_n$; payoff matrix $A \in \mathbb{R}^{m \times n}$.
- Stage-t payoff to row:

$$u(p_t, q_t) \;=\; p_t^\top A q_t.$$

Last-iterate vs time-average

- *Last iterate:* the actual strategy profile at time T,

$$(p_T, q_T).$$

- *Time-average (empirical distribution):*

$$\bar{p}_T := \frac{1}{T}\sum_{t=1}^{T} p_t, \qquad \bar{q}_T := \frac{1}{T}\sum_{t=1}^{T} q_t.$$

- We say *time-average converges* if $(\bar{p}_T, \bar{q}_T)$ approaches the minimax set as $T \to \infty$.
- We say *last iterate converges* if (p_T, q_T) converges to a limit point (e.g., a Nash equilibrium).

Penn
UNIVERSITY of PENNSYLVANIA

Classical guarantees

- Minimax theorem:

$$\max_{p\in\Delta_m}\min_{q\in\Delta_n} p^\top A q \;=\; \min_{q\in\Delta_n}\max_{p\in\Delta_m} p^\top A q \;=:\; v^*.$$

- If both players use no-regret algorithms, the empirical play distribution converges to the set of coarse correlated equilibria; in zero-sum games, $(\bar{p}_T, \bar{q}_T)$ approaches the minimax set and achieves near-optimal value v^*.

- However, recent work on FTRL (Follow-the-Regularized-Leader) dynamics shows a sharp *separation*:
  - Time-averages converge to equilibrium-like objects.
  - The *actual trajectory* (p_t, q_t) can be *recurrent* (Poincaré cycles) and fail to converge.

Penn
UNIVERSITY *of* PENNSYLVANIA

Takeaway

- Many learning algorithms **look like they converge** when we measure averages over time.
- But if we watch what the algorithms are actually *doing each step*, their strategies often **move** rather than settle down.
- These geometric patterns are **not visible** through regret bounds or equilibrium guarantees.

Why This Matters

To truly understand learning in games, we look at *the shape of the path*, not just where the averages end up.

2.2 No-Regret Learning Rules and Known Behaviors

Multiplicative Weights

- Action set $A = \{1, \ldots, k\}$. Maintain weights $w_{t,i} > 0$ and mixed strategy

$$p_t(i) \;=\; \frac{w_{t,i}}{\sum_{j=1}^{k} w_{t,j}}.$$

- At round t, observe a cost (or loss) vector $c_t \in \mathbb{R}^k$.
- Update rule, where $\eta > 0$ is a learning rate:

$$w_{t+1,i} \;=\; w_{t,i} \exp(-\eta\, c_{t,i}), \qquad p_{t+1}(i) \;\propto\; p_t(i) \exp(-\eta\, c_{t,i}),$$

Intuition

- Treat each action like a candidate you are voting for.
- Actions that perform poorly **lose support exponentially fast**.
- Actions that perform well **keep or gain probability**.
- You never fully commit — you keep a soft distribution and shift attention toward winners.
- It is like *putting your money on what worked*, but gently.

Core Idea

MWU is an exponential reward-shifting process. It learns by *rewarding success and punishing failure multiplicatively*.

Online Gradient Descent (OGD)

- Actions $a_t \in K \subset \mathbb{R}^d$ (convex). At round t, observe cost vector c_t with $\|c_t\|_2 \leq C$.
- Unconstrained update:

$$a_{t+1} = a_t - \frac{\eta}{2} c_t.$$

- With constraints, project back to K:

$$a_{t+1} = \Pi_K\Big(a_t - \frac{\eta}{2} c_t\Big).$$

Intuition

- Think of the strategy as a point on a landscape.
- Each round tells you the *direction of steepest improvement*.
- OGD takes a **straight step** downhill in that direction.
- If the step leaves the allowed region (the simplex), we **snap back** to the closest valid point.
- It is like *nudging your strategy a tiny bit toward something better*, every step.

Core Idea

OGD learns by *moving linearly in the direction that reduces loss the fastest*.

Known behaviors in games

- If both players run no-regret algorithms (MWU, OGD, FTRL variants), then:
  - Empirical play (time-average) converges to equilibrium-like sets (Nash/CCE) in zero-sum and more general games.
  - But the *last iterate* often *cycles* around interior equilibria.

- These dynamics can be fast-regret and yet non-convergent in last iterate — a key motivation for studying their geometric trajectories.

2.3 Limitations of Existing Visualization and Analysis Approaches

Regret-based analysis as a “black box”

- Standard theory: if algorithms are no-regret, empirical play converges to CCE / Nash-like objects.
- This guarantees *what averages do*, but not *how the last iterate moves* in state space.
- Convergent, recurrent, and even chaotic systems can all have similarly low regret.

Phase portraits and payoff traces

- Continuous-time dynamics (replicator, FTRL ODEs) can be visualized via 2D phase portraits in simple 2×2 or symmetric games.
- In higher-dimensional, discrete-time repeated games (e.g., 3×3 Shapley), phase portraits become:
  - High-dimensional and hard to project meaningfully.
  - Sensitive to choice of coordinates and projections.
- Payoff vs. time plots only show *one scalar* per round; they lose structural information about the shape of the trajectory.

Why Intensive Principal Component Analysis(InPCA) for Probability Dynamics

The challenge - Strategies are probability distributions. Their natural geometry is curved, not Euclidean.

- Mixed strategies live in a probability simplex

$$\Delta^{k-1} = \{p \in \mathbb{R}^k : p_i \geq 0,\ \sum_i p_i = 1\},$$

  which is not a flat space.
- Changing one probability forces another to adjust — directions are not independent.
- Natural distances between strategies are information-theoretic (KL, Hellinger, Fisher metric), not Euclidean.
- These distances induce a **curved geometric manifold** structure.

Implication

Learning trajectories live on a curved surface. Flattening them with PCA distorts their shape.

What goes wrong with PCA

- PCA assumes data lives in a flat space with straight-line distances.
- Probability vectors live on a **simplex**, where distances behave like KL/Hellinger divergences, not Euclidean norms.
- When we apply PCA to strategy trajectories, it **distorts** the geometry: cycles smear into blobs; polygons collapse into clouds.

What InPCA does

- Uses a kernel derived from the information geometry of distributions.
- Respects **intrinsic distances** between probability states.
- Embeds trajectories into a space where **learning updates appear as clean geometric shapes**.

Experimental Setup

- 3.1 Game Environments
- 3.2 Learning Algorithms
- 3.3 Logging Joint Action Distributions

3.1 Game Environments

Rock–Paper–Scissors (Zero-Sum 3×3)

- Classical benchmark for cycling behavior.
- Unique mixed Nash equilibrium at $(\frac{1}{3}, \frac{1}{3}, \frac{1}{3})$.
- Simple rules, surprisingly rich dynamics.

Shapley's Non-Transitive Game (3×3)

- General-sum payoff matrix with no static dominance ordering.
- Known to generate *limit cycles* under standard learning rules.
- Serves as a stress test for convergence claims.

Shapley's 3×3 Game

Shapley's Game (Shapley, 1964) is a 3×3 general-sum game where each player's best response *cycles*. No action is globally optimal.

$$A_{\text{row}} = \begin{pmatrix} 0 & 1 & -1 \\ -1 & 0 & 1 \\ 1 & -1 & 0 \end{pmatrix}, \qquad A_{\text{col}} = -A_{\text{row}} + \epsilon I$$

Each action is beaten by exactly one other action.

- There exists a unique mixed Nash equilibrium, but **best responses chase each other in cycles**.
- No-regret learning does *not* stabilize — instead, strategies trace persistent orbits.

3.2 Learning Algorithms

Multiplicative Weights Update (MWU)

- Increases probability of actions that performed well.
- Grows and shrinks weights exponentially based on payoff feedback.
- Dynamics tend to be smooth and rotational.

Online Gradient Descent (OGD)

- Moves strategies in the direction that reduces loss.
- Then projects back to the probability simplex.
- Can produce abrupt directional changes due to projection—leading to corner-like turns.

Constant Strategies (Baseline)

- One player never updates their distribution.
- Used to reveal pure-response structure of the adaptive learner.

3.3 Logging Joint Action Distributions

- At each round, we record the **joint mixed strategy**

$$P_t = p_t \otimes q_t \in \mathbb{R}^9,$$

  representing the full distribution over action pairs.
- This lifts the analysis from individual beliefs to their interaction space.
- Each P_t lies on the probability simplex—a curved manifold.
- These logged vectors become the raw material for our geometric embeddings.

Why Joint Distributions? The game is not defined by p_t or q_t alone, but by their *coupled evolution*. Geometry emerges in the joint space.

Visualization Pipeline

- 4.1 Joint Play as a 9-D Probability Vector
- 4.2 Embedding P_t Using InPCA
- 4.3 3D Trajectories Over Time

4.1 Joint Play as a 9-D Probability Vector

Goal: Represent what the two players *jointly do* at each timestep.

- Each player chooses a mixed strategy:

$$p_t \in \Delta^2 \qquad q_t \in \Delta^2,$$

  where Δ^2 is the 3-action simplex (R, P, S).

- The joint play is a probability distribution over action *pairs*:

$$P_t = p_t \otimes q_t \;\in\; \Delta^8 \quad (\text{vector of length } 3 \times 3 = 9).$$

- Entry $P_t(i, j)$ = probability row plays action *i and* column plays action *j*.

Why 9-D?

Pairwise interactions—not individual strategies—determine payoffs, regret, and exploitability. The *outer product* preserves all second-order structure that is invisible in marginal plots of p_t or q_t alone.

4.2 Embedding P_t Using InPCA

Input: Joint distributions $P_t \in \Delta^8$.

1. **Hellinger lift:**

$$S_t = \sqrt{P_t}$$

2. **Center the cloud:**

$$\tilde{S}_t = S_t - \frac{1}{T}\sum_{s=1}^{T} S_s$$

3. **SVD / PCA:**

$$\tilde{S} = U\Sigma V^\top, \qquad X_t = U_{t,1:3} \cdot \frac{\Sigma_{1:3}}{\sqrt{2}}$$

Penn
UNIVERSITY *of* PENNSYLVANIA

How InPCA Finds 3D Geometry

Goal: Reduce a 9-dimensional probability trajectory to 3 meaningful geometric axes.

1. Collect all centered points

$$\tilde{S} \in \mathbb{R}^{T \times 9} = \begin{bmatrix} \tilde{S}_1^\top \\ \tilde{S}_2^\top \\ \vdots \\ \tilde{S}_T^\top \end{bmatrix}$$

Each row is the square-rooted, mean-centered joint distribution at time *t*.

2. Find dominant directions of variation (SVD)

$$\tilde{S} = U\Sigma V^\top$$

- V contains *principal directions* in the 9-D space.
- Σ tells us how important each direction is.
- U gives each timestep's coordinates along those directions.

3. Keep only the top 3 (the "geometric modes")

$$X_t = U_{t,1:3} \cdot \frac{\Sigma_{1:3}}{\sqrt{2}} \quad \in \mathbb{R}^3$$

What this means

We are not plotting raw probabilities. We are plotting the *three strongest ways the strategy profile changes over time*. These axes reveal rings, polygons, and turning points that ordinary PCA and regret curves cannot see.

Interpretation

InPCA embeds probability dynamics into a Euclidean space that respects statistical distance. The resulting 3D coordinates reveal the geometric structure of learning trajectories.

Why InPCA?

Standard PCA treats probabilities like flat vectors. InPCA respects statistical distance, revealing geometric structure that PCA *hides*.

3D Trajectories Over Time

Coordinates: $(X_t^{(1)}, X_t^{(2)}, X_t^{(3)})$ for each timestep t.

- Normalize time:

$$\tau = \frac{t - t_{\min}}{t_{\max} - t_{\min}} \in [0, 1].$$

- Plot trajectories of $(\text{PC1}, \text{PC2}, \text{PC3})$ in 3D.
- Compare “last iterate” vs. “time-average” as separate layers.

Interpretation

The radius, direction, and turns of the trajectory expose *how* algorithms learn—not just *where* they end. Shapes become directly visible.

Penn
UNIVERSITY of PENNSYLVANIA

Why Normalize Time?

- Different runs take the same number of iterations, but *their internal progress is not synchronized*. Early steps explore rapidly; later ones move slowly.
- Raw timestep t is meaningless across runs:

  $t = 150$ may be "early" in one run and "late" in another.

- Normalize time to a common scale:

$$\tau = \frac{t - t_{\min}}{t_{\max} - t_{\min}} \in [0, 1]$$

  so that:
  - $\tau = 0$ always means "beginning of learning",
  - $\tau = 1$ always means "steady state / late behavior",
  - trajectories from different runs become directly comparable.
- This alignment allows us to compute meaningful averages, overlay replicas, and reveal shared geometric patterns.

Penn
University *of* Pennsylvania

Results

- 5.1 MWU vs Constant — Converge Geometry
- 5.2 OGD vs OGD — Hexagonal Cycle
- 5.3 OGD vs MWU — Triangular Cycle
- 5.4 Probability and Geometry

5.1 MWU vs Constant — Converge Geometry

- Setup: row uses MWU, column plays a fixed pure strategy (e.g. always Scissors).
- The joint distribution $P_t = p_t \otimes q_{\text{const}}$ explores a *curved line* in InPCA space.

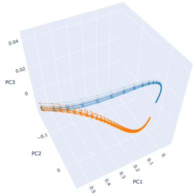

Figure 1: InPCA trajectory for MWU (row) vs constant (col)

5.1 MWU vs Constant — Radius over Time

- Radius in InPCA space:

$$\text{radius}_t = \sqrt{\mathrm{PC1}_t^2 + \mathrm{PC2}_t^2 + \mathrm{PC3}_t^2}.$$

- Last-iterate radius remains roughly stable $\Rightarrow$ persistent cycling.
- Time-average radius decays toward the center $\Rightarrow$ approach to equilibrium in average play.

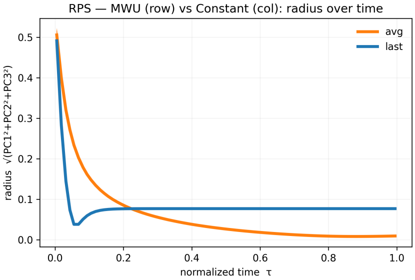

Figure 2: Radius vs. normalized time τ for MWU vs constant.

5.2 OGD vs OGD — Hexagonal Cycle

- Both players run OGD in zero-sum RPS.
- InPCA reveals a *hexagon-like* orbit in the joint distribution P_t.
- Vertices of the hexagon correspond to moments when one player is nearly pure in Rock/Paper/Scissors.
- The path is more “cornered” and piecewise-linear than MWU’s smooth ring, reflecting OGD’s projection step.

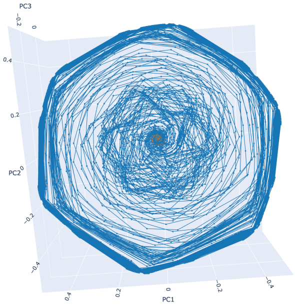

Figure 3: OGD (row) vs OGD (col): hexagon

5.2 OGD vs OGD — Radius and Angular Structure

- Radius over time stays roughly constant for the last iterate $\Rightarrow$ stable limit cycle.
- Angle histogram on the $(\mathrm{PC1}, \mathrm{PC2})$ plane shows a strong $k \approx 6$ harmonic.
- This quantitatively matches the visual hexagon: six preferred directions in the orbit.

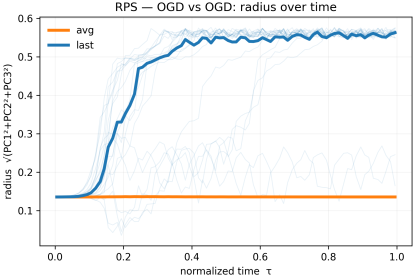

Figure 4: radius vs. time for OGD vs OGD

Penn
University of Pennsylvania

5.3 OGD vs MWU — Triangular Cycle

- Row uses OGD, column uses MWU on RPS.
- InPCA trajectory now concentrates on a *triangle-like* cycle instead of a ring or hexagon.
- Each corner corresponds to OGD's strategy being nearly pure (Rock, Paper, or Scissors).
- MWU's smoother dynamics + OGD's projected updates combine into a three-cornered orbit.

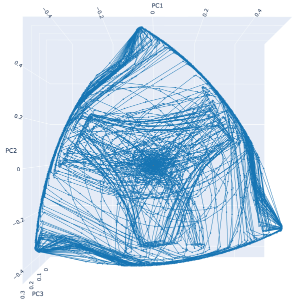

Figure 5: OGD (row) vs MWU (col): triangle

5.3 OGD vs MWU — Radius and Fourier Modes

- Radius for the last iterate again remains roughly constant: a robust limit cycle.
- Angle histogram in the $(\mathrm{PC1}, \mathrm{PC2})$ plane shows a dominant $k \approx 3$ mode.
- Comparing $k \approx 1$ (curve), $k \approx 6$ (hexagon), and $k \approx 3$ (triangle) quantifies the geometry.

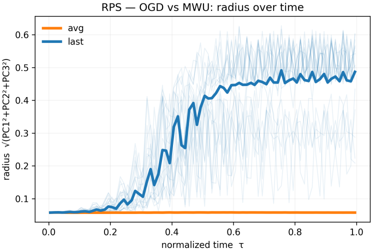

Figure 6: radius vs. time for OGD vs MWU

5.4 Probability and Geometry

- We plot out probabilities, and look at the probabilities at the turning points.
- For OGD, these turning points lie near the *pure strategies*:

$$(1, 0, 0),\ (0, 1, 0),\ (0, 0, 1).$$

- OGD repeatedly drives strategies to the simplex vertices.

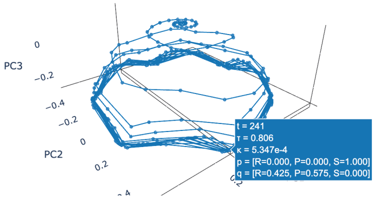

Figure 7: Curvature maxima align with pure actions under OGD

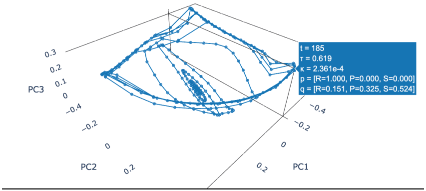

Figure 8: Curvature maxima align with pure actions under OGD for OGD v.s. MWU

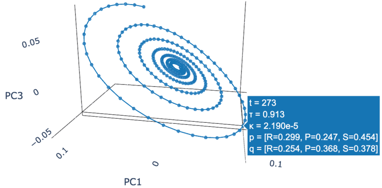

Figure 9: Reference for MWU vs MWU

Why PCA Cannot Reveal Polygonal Geometry

- Standard PCA treats joint distributions P_t as Euclidean vectors. It ignores the **information geometry** of probabilities: the simplex is curved, and distances are not linear.
- Variance along $\sqrt{P_t}$ directions collapses under PCA, flattening distinct modes of motion.
- Thus, PCA answers "*where the mass is*" but not "*how the policy moves*."

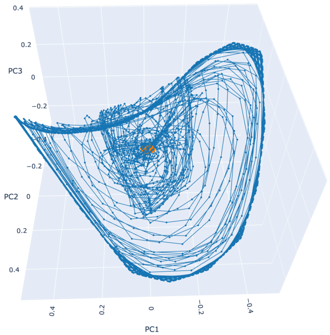

Figure 10: PCA collapses cycles into clouds for OGD vs MWU.

UNIVERSITY of PENNSYLVANIA

Discussion and Future Work

- **Hypothesis: Why OGD Creates Edges**
  - OGD performs *linear* gradient steps, then projects back onto the probability simplex.
  - Projection frequently snaps strategies toward simplex *faces and vertices*, producing *piecewise-linear trajectories*.
  - MWU updates are *multiplicative and smooth*, avoiding hard projections—hence no sharp turns or polygonal edges.

- **Beyond Pairwise Learning**
  - Current pipeline tracks (p_t, q_t) and their outer-product P_t.
  - Future work: log cross-derivatives, regret dynamics, and correlations to uncover *higher-order structure* in learning flows.

- **Algorithmic Extensions**
  - Test other no-regret rules: Follow-The-Regularized-Leader, Optimistic OGD, Replicator Dynamics.
  - Does each algorithm leave a unique "geometric fingerprint" in InPCA space?

References

K. N. Quinn, C. B. Clement, F. De Bernardis, M. D. Niemack, & J. P. Sethna,
Visualizing probabilistic models and data with Intensive Principal Component Analysis,
Proceedings of the National Academy of Sciences (PNAS), 116(28):13762–13767, 2019.
doi: 10.1073/pnas.1817218116.

P. Mertikopoulos, C. Papadimitriou, & G. Piliouras,
Cycles in Adversarial Regularized Learning,
In *Proceedings of the 2018 ACM-SIAM Symposium on Discrete Algorithms (SODA)*,
pp. 2703–2717, SIAM, 2018.
Also available as arXiv:1709.02738.

A. Roth,
Learning in Games (and Games in Learning), Draft textbook, University of Pennsylvania, 2024. provided as course material
Provides definitions and analysis of MWU, OGD, and regret-based dynamics.

Penn
UNIVERSITY of PENNSYLVANIA