Multiplicative Weights and the Hedge Algorithm

Game Theory · Module 9 — Learning In Games

Node 9.4

Position in Knowledge Graph

System: Game Theory **Structural Domain:** Learning in Games **Node:** 9.4

The Multiplicative Weights Update (MWU) method is the workhorse no-regret algorithm: simple, optimal, and ubiquitous. Where regret matching (9.2) sets action probabilities proportional to positive cumulative regrets, MWU sets them proportional to *exponential* weights on cumulative gains — a "soft-max" rather than a "hard-max." This yields the optimal $O(\sqrt{T \ln K})$ external regret bound, the theoretical gold standard. The Hedge algorithm (Freund and Schapire 1997) is the decision-theoretic formulation of MWU: a learner choosing among K "experts" updates weights multiplicatively based on their performance. The same algorithm appears under different names in a dozen fields — Weighted Majority (Littlestone and Warmuth 1994), exponentiated gradient descent (Kivinen and Warmuth 1997), entropic mirror descent, the matrix multiplicative weights method — and each instantiation reveals a different facet of the same structural principle: *downweight losers exponentially, upweight winners exponentially, and the resulting mixture tracks the best fixed expert with sublinear regret.* In the game-theoretic context, MWU is the standard algorithm used to compute Nash equilibria of zero-sum games via the CCE convergence theorem of 9.2.

Formal Definition

Setting. A learner chooses among K actions (or "experts") over T rounds. At round t, the learner picks a distribution $w^t \in \Delta_K$ over actions. Nature (or an adversary) reveals a loss vector $\ell^t \in [0,1]^K$. The learner incurs expected loss $\langle w^t, \ell^t \rangle = \sum_k w_k^t \ell_k^t$.

Multiplicative Weights Update (MWU). Fix a learning rate $\eta > 0$. Initialize $w_k^1 = 1/K$ for all k. Update:

$$\tilde{w}_k^{t+1} \;=\; w_k^t \cdot e^{-\eta \ell_k^t}, \qquad w_k^{t+1} \;=\; \frac{\tilde{w}_k^{t+1}}{\sum_j \tilde{w}_j^{t+1}}.$$

Equivalently, the log-weight of action k decreases linearly in the cumulative loss: $\ln \tilde{w}_k^{t+1} = \ln w_k^1 - \eta \sum_{\tau=1}^{t} \ell_k^\tau$. The normalization ensures the weights form a valid probability distribution.

Hedge algorithm (Freund and Schapire 1997). Identical to MWU with losses in $[-1, 1]$ (or equivalently, gains). The name "Hedge" comes from the financial hedging interpretation: the learner "hedges" their bets across experts.

Weighted Majority (Littlestone and Warmuth 1994). The discrete predecessor: each expert makes a binary prediction, and wrong experts have their weight multiplied by $\beta \in (0, 1)$. The continuous generalization is MWU with $\beta = e^{-\eta}$.

Intuition

Exponential punishment. An action that incurs high loss in a round has its weight multiplied by $e^{-\eta \ell_k^t} < 1$, shrinking it. An action with low loss keeps its weight or grows relative to losers. Over time, the weight distribution concentrates on actions that have performed well historically. The exponential form is key: it penalizes consistently bad actions much faster than linear reweighting would. An action that incurs loss 1 every round has its weight shrink by $e^{-\eta}$ per round — exponential decay, which is why consistently bad actions are pruned quickly.

The tuning knob η. Large η means aggressive learning: bad actions are killed quickly, but the algorithm over-commits and has high regret if the environment shifts. Small η means conservative learning: slow to abandon bad actions, but robust to shifts. The optimal choice $\eta = \sqrt{\ln K / T}$ balances exploration and exploitation, yielding the $O(\sqrt{T \ln K})$ bound.

Entropic regularization. MWU can be derived as "follow the leader" regularized by the negative entropy $\sum_k w_k \ln w_k$. Without regularization, "follow the leader" (play the action with best cumulative performance) is unstable — it can switch abruptly and incur high regret. Adding the entropy penalty keeps the weight distribution from collapsing to a single action, providing the smoothness that enables low regret. This is the "mirror descent" interpretation: MWU is gradient descent in the space of probability distributions, with the KL divergence as the Bregman distance.

Structural Role

9.4 supplies the canonical algorithm for the no-regret framework of 9.2:

- **9.2 establishes the criterion** (sublinear regret ⇒ CCE convergence). 9.4 exhibits the *optimal* algorithm achieving that criterion.
- **9.1 fictitious play** is a hard-max rule (best-respond to empirical distribution). MWU is a soft-max rule (weight by exponential of cumulative gains). The softness is exactly what enables no-regret guarantees.
- **9.3 CFR** uses regret matching, not MWU, at each info set — but MWU could substitute with the same convergence guarantee. The choice of regret matching in CFR is historical and practical (simpler to implement, no learning-rate tuning), not fundamental.
- **9.5 convergence to equilibrium** uses MWU's regret bound as the engine for constructive equilibrium proofs.
- **9.6 reinforcement learning** connects to policy-gradient methods that use entropy regularization, which is the continuous-action analogue of MWU.

Beyond game theory, MWU appears in:

- **Machine learning:** Boosting (AdaBoost is MWU applied to training examples).
- **Online optimization:** Online mirror descent with entropy regularizer.
- **Combinatorial optimization:** The Plotkin–Shmoys–Tardos framework for approximately solving LPs and SDPs using MWU as a separation oracle.
- **Quantum computing:** The matrix MWU method for semidefinite programming (Arora and Kale 2016).

Mathematical Development

The regret bound

Theorem (Freund and Schapire 1997). For MWU with learning rate $\eta \in (0, 1/2]$ and losses $\ell_k^t \in [0, 1]$, the cumulative regret against any fixed action k^* is:

$$\sum_{t=1}^{T} \langle w^t, \ell^t \rangle - \sum_{t=1}^{T} \ell^t_{k^*} \;\le\; \frac{\ln K}{\eta} + \eta T.$$

Proof. Define the potential $\Phi^t = \ln(\sum_k \tilde{w}^t_k)$. We have:

$$\Phi^{t+1} - \Phi^t \;=\; \ln \frac{\sum_k w^t_k e^{-\eta \ell^t_k}}{\sum_k w^t_k}.$$

Using the inequality $e^{-x} \le 1 - x + x^2$ for $x \in [0, 1]$ (valid when $\eta \le 1/2$):

$$\sum_k w^t_k e^{-\eta \ell^t_k} \;\le\; \sum_k w^t_k (1 - \eta \ell^t_k + \eta^2 (\ell^t_k)^2) \;=\; 1 - \eta \langle w^t, \ell^t \rangle + \eta^2 \langle w^t, (\ell^t)^2 \rangle.$$

Taking logarithms and using $\ln(1 - x) \le -x$:

$$\Phi^{t+1} - \Phi^t \;\le\; -\eta \langle w^t, \ell^t \rangle + \eta^2.$$

Summing over $t = 1, \ldots, T$:

$$\Phi^{T+1} - \Phi^1 \;\le\; -\eta \sum_t \langle w^t, \ell^t \rangle + \eta^2 T.$$

Now, $\Phi^1 = \ln K$ (uniform initialization), and $\Phi^{T+1} \ge \ln \tilde{w}^{T+1}_{k^*} = \ln(1/K) - \eta \sum_t \ell^t_{k^*}$ (lower-bounding the sum by any single term). Combining:

$$-\eta \sum_t \ell^t_{k^*} - \ln K \;\le\; -\eta \sum_t \langle w^t, \ell^t \rangle + \eta^2 T + \ln K.$$

Rearranging and dividing by η:

$$\sum_t \langle w^t, \ell^t \rangle - \sum_t \ell^t_{k^*} \;\le\; \frac{2 \ln K}{\eta} + \eta T. \qquad \square$$

(The tighter bound $\ln K/\eta + \eta T$ follows from a sharper use of the exponential inequality.)

Optimal η. Setting $\eta = \sqrt{\ln K / T}$:

$$R^T \;\le\; 2\sqrt{T \ln K}.$$

This is $O(\sqrt{T \ln K})$ and matches the known lower bound up to constants.

MWU for zero-sum game solving

Given a two-player zero-sum game with $m \times n$ payoff matrix A, both players run MWU:

- Row player maintains weights over rows, updated by column player's mixed strategy.
- Column player maintains weights over columns, updated by row player's mixed strategy.

After T rounds, the average mixed strategies $\bar{w}_1^T, \bar{w}_2^T$ satisfy:

$$\text{exploitability} \;\le\; \frac{R_1^T + R_2^T}{T} \;\le\; \frac{4\sqrt{\ln(\max(m,n))}}{\sqrt{T}}.$$

This gives an $O(\ln(mn)/\epsilon^2)$ algorithm for computing ϵ-Nash equilibria of zero-sum games — polynomial in the matrix dimensions and $1/\epsilon$. Compare to fictitious play's rate of $O(1/\epsilon^{m+n-2})$, which is exponentially worse.

Connection to mirror descent

MWU is an instance of **mirror descent** on the probability simplex with the *negative entropy* as the mirror map:

$$\psi(w) \;=\; \sum_k w_k \ln w_k.$$

The Bregman divergence associated with this mirror map is the KL divergence:

$$D_\psi(w\|v) \;=\; \sum_k w_k \ln \frac{w_k}{v_k}.$$

Mirror descent updates: $w^{t+1} = \arg\min_{w\in\Delta_K} \{\eta\langle w, \ell^t\rangle + D_\psi(w\|w^t)\}$. The solution is exactly the MWU update. This derivation shows that MWU is the "natural" gradient descent on the simplex, where "natural" means using the geometry induced by the entropy function rather than the Euclidean distance.

Exponentiated gradient descent

For the online linear optimization variant where the learner picks $w^t \in \Delta_K$ and observes a linear loss $\langle w^t, \ell^t\rangle$, MWU is equivalent to **exponentiated gradient descent** (Kivinen and Warmuth 1997):

$$w_k^{t+1} \;\propto\; w_k^t \cdot \exp\!\left(-\eta\,\frac{\partial}{\partial w_k}\langle w^t, \ell^t\rangle\right) \;=\; w_k^t \cdot e^{-\eta\ell_k^t}.$$

This is the same update as MWU, confirming that MWU is gradient descent in the multiplicative (log) parameterization.

The Weighted Majority algorithm

The discrete version (Littlestone and Warmuth 1994): each expert k makes a binary prediction. The learner follows the weighted majority vote. After each round, multiply the weight of each expert who was wrong by $\beta \in (0,1)$. After T rounds, the number of mistakes is at most:

$$M \;\le\; \frac{\ln K + M^* \ln(1/\beta)}{1-\beta},$$

where M^* is the number of mistakes of the best expert. Setting $\beta = 1/2$: $M \le 2.4(M^* + \ln K)$. This was the precursor to Hedge and MWU.

Worked Example

MWU on a 3x3 zero-sum game

Row player has 3 actions: $\{R, P, S\}$ (Rock, Paper, Scissors). Payoff matrix (Row's gain):

$$A = \begin{pmatrix} 0 & -1 & 1 \\ 1 & 0 & -1 \\ -1 & 1 & 0 \end{pmatrix}.$$

Both run MWU with $\eta = 0.1$. Initialize $w^1 = (1/3, 1/3, 1/3)$ for both.

Round 1. Both play $(1/3, 1/3, 1/3)$. Expected loss for Row against Column's uniform: each action has expected payoff 0 (by symmetry). Loss vector: $\ell^1 = (0, 0, 0)$ (normalized from the payoff). Weights unchanged. Both remain uniform.

This symmetric game is a fixed point for MWU — the Nash equilibrium is reached immediately because the game's symmetry makes every action equally good against uniform play.

Breaking symmetry. Suppose Column deviates to $(0.5, 0.3, 0.2)$ in round 1. Row's loss vector (negated payoff, since we minimize loss): $\ell^1_R = -(0{\cdot}0.5+(-1){\cdot}0.3+1{\cdot}0.2) = 0.1$, $\ell^1_P = -(1{\cdot}0.5+0{\cdot}0.3+(-1){\cdot}0.2) = -0.3$, $\ell^1_S = -((-1) \cdot 0.5 + 1 \cdot 0.3 + 0 \cdot 0.2) = 0.2$.

Updated weights: $\tilde{w}^2_R = (1/3)e^{-0.1 \cdot 0.1} = 0.3300$, $\tilde{w}^2_P = (1/3)e^{0.1 \cdot 0.3} = 0.3434$, $\tilde{w}^2_S = (1/3)e^{-0.1 \cdot 0.2} = 0.3267$. After normalization: $w^2 \approx (0.330, 0.343, 0.327)$. Row shifts toward Paper (which exploits Column's overweighting of Rock). Over subsequent rounds, both players adjust, and the average strategies converge to the Nash $(1/3, 1/3, 1/3)$ from any initial condition.

Convergence rate comparison

For the 3x3 RPS game ($K = 3$), after T rounds:

T	MWU exploitability bound	Fictitious play exploitability bound
100	$2\sqrt{\ln 3/100} \approx 0.21$	$O(100^{-1/4}) \approx 0.32$
10,000	$2\sqrt{\ln 3/10000} \approx 0.021$	$O(10000^{-1/4}) \approx 0.10$

MWU is faster by a substantial margin, and the gap widens with game size.

Poker Anchor

Concept mapping

Formal object	Poker instantiation
Expert k	An action at a decision point (fold, call, raise 50%, raise 100%, etc.)
Weight w^t_k	The probability assigned to action k at iteration t of a solver
Loss ℓ^t_k	The negative counterfactual value of action k at this info set, this iteration
Learning rate η	Controls how aggressively the solver shifts between actions across iterations
Average strategy $\bar{w}^T$	The solver's GTO output at this info set

Concrete situation: MWU vs regret matching in solvers

Most poker solvers use regret matching rather than MWU at each info set. The reason is practical: regret matching requires no learning-rate parameter (it is parameter-free), while MWU requires choosing η, which depends on the time horizon T — a quantity not always known in advance. Adaptive learning-rate variants of MWU (e.g., AdaHedge) exist but add complexity. Regret matching achieves slightly worse theoretical bounds ($O(\sqrt{T})$ vs $O(\sqrt{T \ln K})$), but in practice the difference is negligible because the number of actions at each info set is small (typically 2-5).

However, MWU's theoretical optimality matters for understanding *why* the $O(T^{-1/2})$ convergence rate is fundamental. No algorithm — not regret matching, not MWU, not any future invention — can beat $\Omega(\sqrt{T})$ regret against adversarial losses, so the $O(T^{-1/2})$ exploitability rate is a hard lower bound on how fast any self-play method can converge to Nash.

What this understanding buys you

The learning rate controls the speed-stability tradeoff. In solver tuning, if a solver offers MWU-based updates, a higher η converges faster initially but may oscillate; a lower η is more stable but slower. Understanding MWU tells you this is not a bug — it is a fundamental tradeoff, and the optimal η depends on how many iterations you plan to run.

MWU explains why solver outputs are smooth. The exponential weighting ensures that no action is ever assigned probability exactly 0 until it has been consistently dominated for many iterations. This smoothness is why solver strategies involve many mixed frequencies rather than sharp pure-strategy prescriptions — the no-regret machinery produces genuinely mixed strategies as output, not approximations to pure ones.

Cross-Links

- **9.1 Fictitious play** — hard-max predecessor; MWU is the soft-max improvement.
- **9.2 No-regret learning** — MWU achieves the optimal bound within this framework.
- **9.3 CFR** — uses regret matching instead of MWU; the same convergence theory applies.
- **9.5 Convergence to equilibrium** — MWU's rate is the engine for quantitative convergence results.
- **Online convex optimization** — MWU is mirror descent with entropy regularizer.
- **Machine learning: boosting** — AdaBoost is MWU over training examples.
- **Optimization: LP/SDP solving** — MWU as separation oracle (Plotkin-Shmoys-Tardos, Arora-Kale).

Structural Compression

Invariant: Multiplicative (exponential) reweighting of actions based on cumulative losses achieves $O(\sqrt{T \ln K})$ external regret — optimal among all online learning algorithms in the adversarial full-information setting — and when used by all players in a zero-sum game, computes an ϵ-Nash equilibrium in $O(\ln K/\epsilon^2)$ iterations.

Mechanism: Each action's weight is multiplied by $e^{-\eta \cdot \text{loss}}$ per round and renormalized. The potential function $\Phi^t = \ln \sum_k \tilde{w}_k^t$ telescopes to give the regret bound. The learning rate $\eta = \sqrt{\ln K/T}$ optimally balances the two terms.

Cross-System Echo: MWU is the discrete analogue of **gradient flow on the probability simplex with KL-divergence geometry** (the "information geometry" of Amari). In statistical mechanics, the Gibbs distribution $p_k \propto e^{-\beta E_k}$ assigns probabilities proportional to $e^{-\text{energy}}$, which is exactly the MWU steady state when losses are constant. In biology, the replicator dynamic — the continuous-time version of evolutionary selection — is the continuous-time limit of MWU (Taylor and Jonker 1978, Losert and Akin 1983). The exponential reweighting principle — penalize bad states exponentially, favor good states exponentially — is one of the most universal computational principles in nature and mathematics.

Exercises

1. Prove the MWU regret bound $R^T \leq \ln K/\eta + \eta T$ from the potential function argument. Verify the optimal $\eta = \sqrt{\ln K/T}$ yields $R^T \leq 2\sqrt{T \ln K}$.
2. Implement MWU for a 3x3 zero-sum game (both players). Run for 10,000 iterations and plot the exploitability as a function of T. Verify the $O(T^{-1/2})$ rate empirically.
3. Show that the Weighted Majority algorithm is a special case of MWU with binary losses. Derive the mistake bound $M \leq 2.4(M^* + \ln K)$ from the general regret bound.
4. Derive the MWU update as the solution to the mirror-descent optimization problem: $w^{t+1} = \arg\min_{w \in \Delta_K} \{\eta \langle w, \ell^t \rangle + D_{KL}(w \| w^t)\}$. (Hint: use Lagrange multipliers on the simplex constraint.)
5. Compare MWU and regret matching on a 2-action game with adversarial losses that oscillate between $(0, 1)$ and $(1, 0)$. Which converges faster? Which has smoother weight trajectories? Why?
6. Show that AdaBoost can be derived as MWU applied to a game between a learner (choosing among weak classifiers) and an adversary (choosing training examples). What is the payoff matrix? What is the equilibrium interpretation of the boosted classifier?
7. **Argue, structurally,** why exponential reweighting (MWU) achieves better regret than linear reweighting (fictitious play) in general, but both converge in zero-sum games. What structural property of zero-sum games makes linear reweighting sufficient, and what do general games demand that only exponential reweighting provides?

Game Theory · Module 9 — Learning In Games · Node 9.4
Concepts: convergence, stochastic-optimization, cost-function
Prerequisites: 9.1, 9.2
Enables: 9.5, 9.6

hbar.university — own.your.cognition