Mathematical Modeling and Algorithm Application
ISSN: 3006-0842 | Vol. 10, No. 1, 2026

Multiplicative Weights for Online Binary Prediction under Stable and Changing Expert Reliability: A Simulation Study

Hengzhi Shao

University of California, Santa Barbara, Santa Barbara, CA 93106, United States

Abstract: Combining advisers with unknown and changing reliability is a central problem in online prediction. This paper studies a deterministic exponential weighted-majority predictor in two synthetic environments. Nine experts forecast 1,000 balanced binary outcomes, and 200 Monte Carlo repetitions compared six learning rates from 0.01 to 1.00, equal majority voting, the best fixed expert, and a two-phase oracle. With stable reliabilities, mean errors declined from 241.62 at learning rate 0.01 to 203.10 at learning rate 1.00, close to the best fixed expert's 200.04 and well below equal majority's 407.34. When the initially strongest and weakest experts exchanged accuracies at round 501, mean errors declined from 315.56 to 282.57 as the learning rate increased, but the latter result still incurred 82.34 more errors than the phase-specific oracle. Short-run adaptation showed the opposite learning-rate pattern: during rounds 501-600, learning rate 0.01 averaged 45.11 errors, whereas rates from 0.05 to 1.00 averaged about 63. Faster updating therefore improved full-horizon performance but created greater vulnerability immediately after an abrupt reversal. Static regret alone understated the cost of changing reliability.

Keywords: Online learning; Multiplicative weights; Expert advice; Nonstationarity; Regret

1. Introduction

Many sequential decisions depend on several forecasts rather than a single source. Examples include combining statistical models, human judgments, sensors, or trading rules. At each round, a decision-maker receives advice, chooses a prediction, observes the outcome, and then learns which advisers were correct. The central difficulty is that reliability is unknown in advance and may not remain constant. Equal voting is simple, but it treats accurate and inaccurate advisers identically. An online method should instead use observed performance while avoiding an irreversible commitment based on a short early sequence.

The weighted-majority framework assigns a positive weight to each expert and discounts experts that make mistakes [1]. Exponential update rules belong to the broader multiplicative-weights or Hedge family [2,3]. Closely related randomized and loss-based variants admit guarantees relative to the best fixed expert in hindsight [4,5]. Because the present experiment uses a deterministic thresholded vote, its regret values are empirical diagnostics rather than a new theoretical guarantee. The same update principle also appears in boosting, optimization, and game-theoretic algorithms [2,3].

Two design choices motivate the experiment. First, the learning rate controls how sharply weights respond to mistakes. A very small value keeps a broad mixture but may learn too slowly, whereas a large value concentrates quickly on the apparent leader. Second, comparison with the best fixed expert can conceal poor adaptation when the identity of the best expert changes. Abrupt change can turn previously useful evidence into inertia. Tracking algorithms address this problem by sharing, forgetting, or restarting accumulated weights [6], but the behavior of the unmodified rule must be established first.

This study addresses two questions: how does the learning rate affect cumulative error under stable expert reliabilities, and how well does the unmodified algorithm respond when the most reliable expert changes? The initial expectation was that a moderate rate would balance slow learning against excessive concentration, and that a reliability switch would create a much larger gap to a switching oracle than to the best fixed expert. A controlled simulation tests six learning rates in stable and changing environments and compares the algorithm with equal majority voting, the best fixed expert, and a phase-specific oracle.

2. Literature Review

2.1. Prediction with expert advice

Littlestone and Warmuth introduced the weighted majority algorithm as a method for combining a pool of prediction algorithms and bounding the compound learner's mistakes relative to a good member of the pool [1]. Freund and Schapire extended the multiplicative allocation idea to a general online decision problem and connected it to boosting [2]. Cesa-Bianchi et al. developed a detailed worst-case analysis of binary prediction with expert advice, measuring performance through the difference between the learner's mistakes and those of the best expert [4]. These works established that no probabilistic model of the outcome sequence is required for the central regret comparison.

Later treatments placed expert advice within a wider theory of prediction and learning. Cesa-Bianchi and Lugosi presented individual-sequence prediction as a framework linking machine learning, information theory, statistics, and repeated games [5]. Arora, Hazan, and Kale showed that many apparently different algorithms share the same structure: maintain a distribution, observe losses, and update each coordinate multiplicatively [3]. This perspective motivates the normalized exponential update used in the present simulation.

2.2. Learning rates and changing reliability

The learning rate determines the concentration of the expert

distribution. Fixed rates are easy to implement but require a trade-off between initial adaptation and stability. AdaHedge tunes this response from the observed difficulty of the sequence [7]. Second-order and parameter-free methods either combine multiple rates or avoid selecting a single prespecified rate [8,9], while analysis of decreasing-rate Hedge clarifies when a simpler schedule adapts to stochastic gaps [10]. The current study does not introduce a new tuning rule; instead, it compares a small grid of fixed rates so that the practical sensitivity of the basic algorithm can be observed directly.

Nonstationarity creates a different issue. Herbster and Warmuth formalized tracking the best expert when the comparator is allowed to switch and proposed sharing weight between experts to prevent the learner from becoming trapped by old evidence [6]. Later work connected fixed share with mirror descent and developed interval-based or parameter-free guarantees for changing environments [11-14]. The distinction is essential: static regret compares with one expert over the full horizon, while a switching benchmark compares with a sequence of experts. The simulation below reports both quantities and intentionally leaves the standard cumulative update unchanged.

2.3. Empirical comparisons and applications

Empirical studies show that aggregation performance depends on the loss, the prediction task, and the composition of the expert pool. Weighted-majority variants have been evaluated in calendar scheduling [15], and adaptive aggregators have been compared on expert forecasts of National Football League games [16]. Sequential expert aggregation has also been applied to electricity-consumption forecasting with specialized experts [17] and to probabilistic load and price forecasting in GEFCom2014 [18]. These comparisons motivate the use of multiple baselines and learning rates here, while the synthetic design isolates the effect of reliability change from application-specific preprocessing.

3. Method

3.1. Online prediction model

The experiment used binary outcomes. On round t, each of N experts issued a prediction x(t, i) in {0,1}. The learner then predicted, after which the true outcome y(t) was revealed. Expert i received zero-one loss l(t, i), equal to 1 when x(t, i) differed from y(t) and 0 otherwise. All experts began with equal normalized weight q(1, i)=1/N. The deterministic learner predicted one when the weighted vote was at least one half:

$$\hat{y}t = 1\{\sum i=1N\ qt, i\ xt, i \geq 0.5\}. \quad (1)$$

After observing the outcome, each expert's weight received an exponential penalty and the weights were renormalized. With learning rate $\eta>0$, the update was

$$qt+1, i = qt, i \exp(-\eta lt, i) / [\sum j=1N\ qt, j \exp(-\eta lt, j)]. \quad (2)$$

A correct expert retained its unnormalized weight, while an incorrect expert was multiplied by $\exp(-\eta)$. Renormalizing after every round kept the weights summing to one, improved numerical scaling, and made each weight directly interpretable as a share of the vote. The weighted vote should not be read as a calibrated probability. No expert was randomly sampled; the final prediction always used the deterministic 0.5 threshold.

3.2. Simulation environments

Each simulated sequence contained 1,000 rounds and nine experts. Outcomes were independent Bernoulli(0.5) draws. Conditional on the outcome, expert i was correct with a prespecified probability, independently across experts and rounds. Stable accuracies were 0.80, 0.72, 0.65, 0.58, 0.52, 0.48, 0.42, 0.35, and 0.30. This range made equal treatment clearly suboptimal while retaining experts near, above, and below chance.

The changing environment used the same accuracies in rounds 1-500. At round 501, Expert 1 changed from 0.80 to 0.30 and Expert 9 changed from 0.30 to 0.80; all other experts remained unchanged. Expert 2 therefore supplied a stable accuracy of 0.72 over the full horizon. This design separates the ability to identify a strong fixed expert from the ability to recover a newly strong expert whose earlier record was poor.

Six learning rates were evaluated: 0.01, 0.05, 0.10, 0.20, 0.50, and 1.00. Each environment was repeated 200 times. The stable and changing experiments used NumPy seeds 20260719 and 20260720, respectively. Within each repetition, every learning rate received exactly the same outcomes and expert predictions, so rate comparisons were paired by design. The complete source code is provided in the accompanying archive mwa_reproducibility_package.zip, which contains mwa_experiment.py, a pinned requirements.txt, and the generated tables and figures. The code was run with Python 3.12.13, NumPy 2.3.5, and Matplotlib 3.10.8. From the project directory, the command python mwa_experiment.py recreates all outputs in output/mwa_experiment; no external data or model training are required. The data-generating mechanism, methods, performance measures, and Monte Carlo repetitions are reported in line with general guidance for simulation studies [19].

3.3. Baselines and evaluation measures

Equal majority voting predicted one when at least five of the nine experts predicted one. The best fixed expert was the single expert with the smallest total loss over all 1,000 rounds. In the changing environment, a phase-specific oracle selected the lowest-loss expert separately in rounds 1-500 and 501-1,000. Both expert benchmarks use hindsight and are not prospective forecasting rules; they diagnose the cost of learning and the additional cost of the reliability change.

For each run, the learner's cumulative loss L(MW) was its total number of zero-one errors. Static regret compared this loss with the best fixed expert:

$$Rstatic = LMW - mini\ Li. \quad (3)$$

Let L(i,1) and L(i,2) denote expert i's loss in the first and second 500-round phases. Switching regret used the best expert separately in each phase:

$$Rswitch = LMW - [mini\ Li(1) + mini\ Li(2)]. \quad (4)$$

Accuracy, normalized expert weights, and errors in rounds 501-600 were also recorded. This 100-round post-change window measures immediate exposure to the reliability reversal. Tables report Monte Carlo means and sample standard deviations across 200 repetitions.

4. Results

4.1. Stable expert reliability

Under stable reliability, increasing the learning rate from 0.01 to 0.05 produced the largest improvement (Table 1). Mean cumulative errors fell from 241.62 to 208.60, then plateaued at 203.57 for rate 0.20 and 203.10 for rate 1.00. The

latter accuracy was 0.7969. The best fixed expert averaged 200.04 errors, giving static regret of 3.06 at rate 1.00, while equal majority averaged 407.34 errors. Most of the gain therefore came from discounting weak experts; additional concentration beyond rate 0.20 rarely changed the final binary vote.

Table 1. Performance under stable expert reliability.

Learning rate	Cumulative errors	Accuracy	Static regret
0.01	241.62 (13.75)	0.7584	41.58 (10.91)
0.05	208.60 (13.47)	0.7914	8.56 (5.30)
0.10	204.90 (13.28)	0.7951	4.86 (3.79)
0.20	203.57 (13.24)	0.7964	3.54 (2.49)
0.50	203.24 (13.57)	0.7968	3.19 (2.12)
1.00	203.10 (13.52)	0.7969	3.06 (1.72)

Values are means across 200 repetitions; standard deviations are in parentheses. Best fixed expert: 200.04 (12.95) errors; equal majority: 407.34 (15.74).

4.2. Changing expert reliability

Under changing reliability, full-horizon errors again fell as the learning rate increased: 315.56 at rate 0.01 versus 282.57 at rate 1.00 (Table 2). At rate 1.00, static regret was only 3.27, but switching regret was 82.34 because the phase-specific oracle averaged 200.24 errors. The first 100 post-change rounds showed the opposite learning-rate ranking: rate 0.01 made 45.11 (5.70) errors, whereas all larger rates made about 63. The smaller rate remained more diversified and was less exposed to the abrupt reversal, although its slow pre-change learning kept its total error highest.

Table 2. Performance when the strongest and weakest experts exchange reliability.

Learning rate	Cumulative errors	Accuracy	Static regret	Switching regret	Errors, rounds 501-600
0.01	315.56 (16.26)	0.6844	36.25 (11.67)	115.32 (18.21)	45.11 (5.70)
0.05	287.00 (14.96)	0.7130	7.69 (5.60)	86.76 (19.14)	63.12 (8.12)
0.10	284.45 (13.76)	0.7155	5.15 (3.62)	84.22 (18.11)	63.70 (8.04)
0.20	283.05 (13.70)	0.7170	3.74 (2.50)	82.81 (17.86)	63.72 (8.04)
0.50	282.71 (13.76)	0.7173	3.41 (1.94)	82.48 (17.87)	63.72 (8.03)
1.00	282.57 (13.78)	0.7174	3.27 (1.78)	82.34 (17.79)	63.61 (8.06)

Values are means across 200 repetitions; standard deviations are in parentheses. The final column counts errors in rounds 501-600. Equal majority: 407.82 (15.89); best fixed expert: 279.31 (13.65); phase-specific oracle: 200.24 (11.87).

Figure 1 displays the mean cumulative error paths. In the stable panel, rates 0.20 and 1.00 nearly coincided with the best fixed expert, while rate 0.01 retained a visible learning cost. In the changing panel, the high-rate curves steepened immediately after round 500 before settling near the best fixed expert; rate 0.01 entered the switch with more accumulated error but a smaller local increase. All multiplicative-weights curves remained well above the phase-specific oracle after the switch.

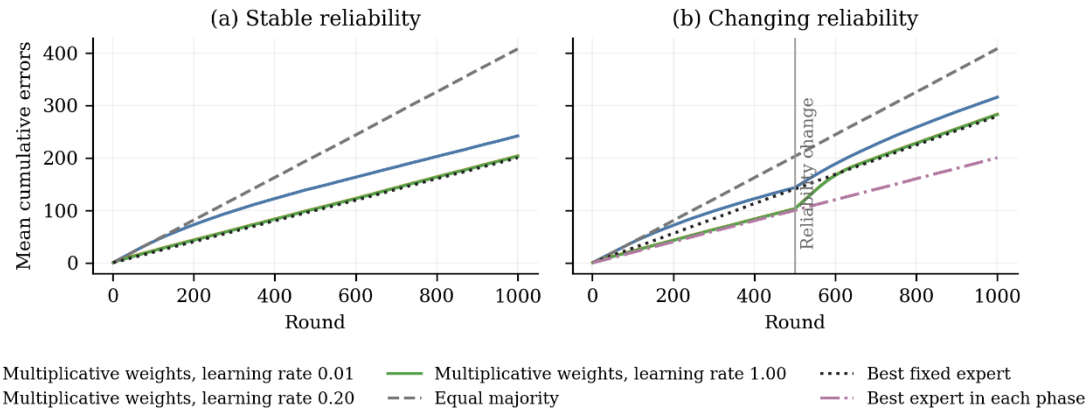


Figure 1. Mean cumulative errors across 200 repetitions. The vertical line marks the reliability change at round 500. The phase-specific oracle is shown only for the changing environment.

The mean weight paths at learning rate 0.20 explain this benchmark gap (Figure 2). Before the change, Expert 1 acquired almost all weight because its accuracy was 0.80. After its accuracy fell to 0.30, its weight declined, but Expert 9 did not recover from the penalty accumulated during the first 500 rounds. Expert 2, whose accuracy remained 0.72, instead became dominant near round 600. The algorithm adapted away from a failing expert but moved toward the best full-horizon fixed alternative rather than the best expert in the new phase.

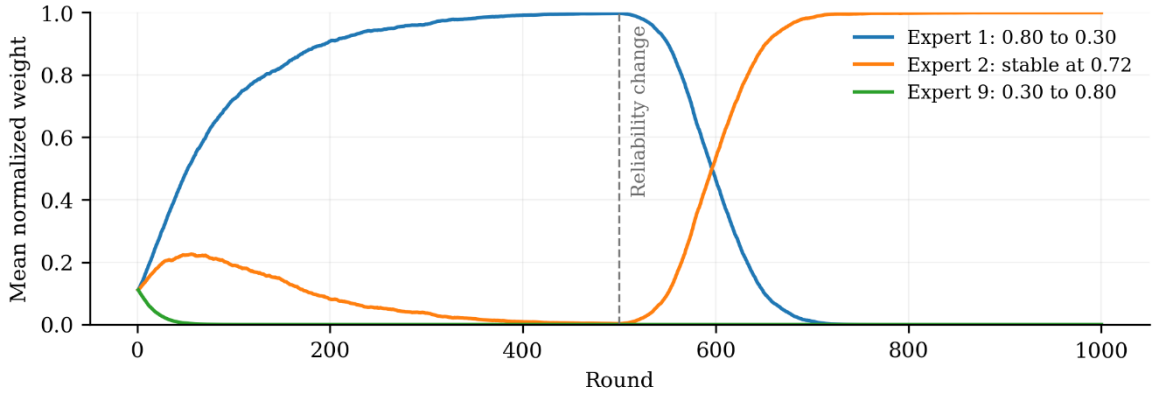


Figure 2. Mean normalized weights in the changing environment at learning rate 0.20. Only the two switching experts and the strongest stable expert are shown.

5. Discussion

5.1. Learning-rate sensitivity

Learning rate affected stable efficiency and shock robustness differently. Under stable reliability, rate 0.01 discounted mistakes so gently that much of the horizon was spent identifying a reliability gap already present in the data-generating process; rates at or above 0.20 produced similar final totals. In the changing environment, larger rates also improved full-horizon totals, but the first 100 post-change rounds reversed that ranking. Rate 0.01 retained a diversified vote and averaged 45.11 errors, whereas rates from 0.05 to 1.00, which had concentrated strongly on Expert 1, averaged roughly 63.

This pattern is a stability-adaptation trade-off. Slow updating imposed a large cost before the change but cushioned the immediate reversal; fast updating learned the stable hierarchy efficiently but had more accumulated evidence to undo. The initial expectation of deterioration at very large rates was therefore only partly supported: rates 0.50 and 1.00 did not worsen final totals in this controlled setting, yet their short-run post-change performance was poor. The independent errors, clear reliability gaps, and long horizon favored rapid concentration. Adaptive or parameter-free procedures become important when persistence and change cannot be known in advance [7-10].

5.2. Static regret and adaptation to change

The central result is the contrast between static and switching regret. At the larger learning rates, the changing-environment predictor was within roughly three errors of the best fixed expert. That comparison correctly shows that the algorithm found a strong single source over the full horizon. Nevertheless, it made about 82 more errors than the oracle that chose a different expert in each half. Low static regret therefore did not establish that the learner followed the expert who was currently most reliable.

The weight trajectories show why changing the learning rate cannot solve this structural problem by itself. Standard cumulative updating stores the entire loss history. Expert 9 became the most accurate source after round 500, but its poor first-half record had already driven its weight close to zero. Expert 2 had the best full-horizon record and absorbed the weight released by Expert 1. Fixed-share and strongly adaptive methods preserve, reintroduce, or redistribute weight so that previously poor experts can return [6,11-14]. Restarts, sliding loss windows, or explicit change detection provide related alternatives for future experiments.

5.3. Limitations

This study uses synthetic data to isolate algorithm behavior, so it does not estimate performance in a specific application. Outcomes were balanced, expert errors were conditionally independent, and only one abrupt midpoint change was tested. Real experts may make correlated mistakes, drift gradually, or react to the learner. The deterministic thresholded vote also differs from randomized Hedge variants used in some regret analyses. Finally, the experiment considered six fixed learning rates, one reliability pattern, and one 100-round adaptation window. The reported standard deviations quantify Monte Carlo variation under these assumptions; they do not establish external validity.

6. Conclusion

This simulation shows that deterministic multiplicative-weights voting can learn a stable expert hierarchy effectively. Under stable reliability, rates from 0.20 to 1.00 approached the best fixed expert and strongly outperformed equal majority voting. The abrupt switch exposed a more complex trade-off: large rates achieved the lowest full-horizon error but averaged about 63 errors in the first 100 post-change rounds and retained switching regret above 82; rate 0.01 made only 45.11 immediate post-change errors but performed worse overall. The weights ultimately moved toward stable Expert 2 rather than recovering newly reliable Expert 9. Learning rate therefore controls the balance between early learning and shock robustness, but it cannot erase cumulative memory. Low static regret should not be interpreted as rapid adaptation to structural change. Future work should compare fixed-share, restart, and sliding-window variants under

gradual and correlated reliability shifts.

References

[1] Littlestone, N., & Warmuth, M. K. (1994). The weighted majority algorithm. Information and Computation, 108(2), 212–261. https://doi.org/10.1006/inco.1994.1009

[2] Freund, Y., & Schapire, R. E. (1997). A decision-theoretic generalization of on-line learning and an application to boosting. Journal of Computer and System Sciences, 55(1), 119–139. https://doi.org/10.1006/jcss.1997.1504

[3] Arora, S., Hazan, E., & Kale, S. (2012). The multiplicative weights update method: a meta-algorithm and applications. Theory of Computing, 8(1), 121–164. https://doi.org/10.4086/toc.2012.v008a006

[4] Cesa-Bianchi, N., Freund, Y., Haussler, D., Helmbold, D. P., Schapire, R. E., & Warmuth, M. K. (1997). How to use expert advice. Journal of the ACM, 44(3), 427–485. https://doi.org/10.1145/258128.258179

[5] Cesa-Bianchi, N., & Lugosi, G. (2006). Prediction, learning, and games. Cambridge University Press. https://doi.org/10.1017/CBO9780511546921

[6] Herbster, M., & Warmuth, M. K. (1998). Tracking the best expert. Machine Learning, 32(2), 151–178. https://doi.org/10.1023/A:1007424614876

[7] de Rooij, S., van Erven, T., Grünwald, P. D., & Koolen, W. M. (2014). Follow the leader if you can, Hedge if you must. Journal of Machine Learning Research, 15, 1281–1316. https://jmlr.org/papers/v15/rooij14a.html

[8] Koolen, W. M., & Van Erven, T. (2015). Second-order quantile methods for experts and combinatorial games. Proceedings of the 28th Conference on Learning Theory, 40, 1155–1175. https://proceedings.mlr.press/v40/Koolen15a.html

[9] Luo, H., & Schapire, R. E. (2015). Achieving all with no parameters: AdaNormalHedge. Proceedings of the 28th Conference on Learning Theory, 40, 1286–1304. https://proceedings.mlr.press/v40/Luo15.html

[10] Mourtada, J., & Gaïffas, S. (2019). On the optimality of the Hedge algorithm in the stochastic regime. Journal of Machine Learning Research, 20, 1–28. https://jmlr.org/papers/v20/18-869.html

[11] Cesa-Bianchi, N., Gaillard, P., Lugosi, G., & Stoltz, G. (2012). Mirror descent meets fixed share (and feels no regret). Advances in Neural Information Processing Systems, 25, 989–997. https://proceedings.neurips.cc/paper/2012/hash/8e6b42f1644ecb1327dc03ab345e618b-Abstract.html

[12] Monteleoni, C., & Jaakkola, T. S. (2003). Online learning of non-stationary sequences. Advances in Neural Information Processing Systems, 16, 1093–1100. https://proceedings.neurips.cc/paper/2003/hash/feecee9f1643651799ede2740927317a-Abstract.html

[13] Daniely, A., Gonen, A., & Shalev-Shwartz, S. (2015). Strongly adaptive online learning. Proceedings of the 32nd International Conference on Machine Learning, 37, 1405–1411. https://proceedings.mlr.press/v37/daniely15.html

[14] Jun, K.-S., Orabona, F., Wright, S., & Willett, R. (2017). Improved strongly adaptive online learning using coin betting. Proceedings of the 20th International Conference on Artificial Intelligence and Statistics, 54, 943–951. https://proceedings.mlr.press/v54/jun17a.html

[15] Blum, A. (1997). Empirical support for Winnow and Weighted-Majority algorithms: results on a calendar scheduling domain. Machine Learning, 26(1), 5–23. https://doi.org/10.1023/A:1007335615132

[16] Dani, V., Madani, O., Pennock, D. M., Sanghai, S., & Galebach, B. (2006). An empirical comparison of algorithms for aggregating expert predictions. Proceedings of the 22nd Conference on Uncertainty in Artificial Intelligence, 106–113. https://dl.acm.org/doi/10.5555/3020419.3020433

[17] Devaine, M., Gaillard, P., Goude, Y., & Stoltz, G. (2013). Forecasting electricity consumption by aggregating specialized experts. Machine Learning, 90(2), 231–260. https://doi.org/10.1007/s10994-012-5314-7

[18] Gaillard, P., Goude, Y., & Nedellec, R. (2016). Additive models and robust aggregation for GEFCom2014 probabilistic electric load and electricity price forecasting. International Journal of Forecasting, 32(3), 1038–1050. https://doi.org/10.1016/j.ijforecast.2015.12.001

[19] Morris, T. P., White, I. R., & Crowther, M. J. (2019). Using simulation studies to evaluate statistical methods. Statistics in Medicine, 38(11), 2074–2102. https://doi.org/10.1002/sim.8086