

When Reproducible Revenue-Management Simulations Are Still Invalid: Evidence From a Continuation-Value Policy Audit

ZHAORUI ZHU

CEO@ASKZHU.AI

ASKZHU.AI

Research Article

Keywords: airline revenue management, common random numbers, continuation value, EMSR-b, holdout validation, reproducibility

Posted Date: September 22nd, 2026

DOI: <https://doi.org/10.21203/rs.3.rs-10976301/v1>

License: © ⓘ This work is licensed under a Creative Commons Attribution 4.0 International License.

[Read Full License](#)

Additional Declarations: No competing interests reported.

Abstract

Simulation code can regenerate identical outputs while still supporting invalid policy conclusions. We audit an airline revenue-management continuation-value heuristic against EMSR-b-style seat protection across 5,100 synthetic scenarios and 45,900 policy evaluations. Four failures emerge: misaligned stochastic streams, pre-decision continuation values, degenerate percentage-error states, and severe instance-level holdout failure. After strict common-random-number correction, the original heuristic underperforms EMSR-b by 15.84%; a distilled approximation narrows the gap to 3.26% but does not establish superiority. We translate these findings into a ten-stage validation protocol for researchers, airlines, and purchasers of algorithmic revenue-management systems.

1. INTRODUCTION

Airline revenue management allocates perishable seat capacity across temporally separated and price-sensitive demand, with recent work extending to choice-based offer optimization [20]. Because a seat sold today cannot be recovered for a later high-yield request, both seat-protection and dynamic-pricing methods depend on an estimate of the marginal value of remaining capacity. EMSR-based controls provide a widely used probabilistic approximation for nested fare classes, whereas dynamic programming provides a conceptual benchmark when the state and action spaces are sufficiently small [1]–[4].

For airline revenue-management (RM) practitioners, this distinction is not merely academic. Airlines increasingly evaluate third-party dynamic pricing engines, approximate dynamic programming controllers, and machine-learning-based inventory optimizers through vendor-provided simulation studies. A simulation that appears to demonstrate double-digit revenue gains can influence costly licensing decisions, pilot commitments, and internal resource allocation—yet if the comparison is methodologically flawed, those decisions rest on invalid evidence. The risk is compounded by the fact that RM simulation environments are complex: multiple fare classes, non-stationary demand, stochastic arrivals, and nested inventory controls create many opportunities for subtle design errors that nonetheless produce clean, reproducible-looking output tables.

Empirical evaluation is difficult because real airline transaction and availability data are commercially sensitive and policy changes affect the demand that is subsequently observed. Synthetic simulation is therefore common in early-stage algorithm research. Yet simulation creates a second identification problem: a reported difference may reflect the policy, the stochastic realization assigned to it, or an implementation choice hidden inside the simulator. A program can deterministically regenerate the same CSV and still fail to support the causal interpretation stated in the manuscript.

This paper examines that distinction through an audit of a finite-horizon pricing experiment. The archived experiment compared an EMSR-b-style benchmark with a price-adjustment heuristic using a one-step continuation estimate, an inventory-threshold availability rule, an aggregate future-demand look-ahead, and combinations of these components. A later extension fitted a polynomial value approximator to

small dynamic-programming instances and reported near-benchmark policy revenue. The present study asks whether those conclusions survive four stress tests: common exogenous randomness, correct separation of pre- and post-decision states, instance-level holdout validation, and non-degenerate error reporting.

The audit produces three substantive findings. First, the archived outputs are mechanically reproducible, but the policy loop advances one random number generator sequentially across policies; policy order therefore determines the demand stream assigned to each method. Second, the learned value function fails on complete held-out instances despite a high in-sample R^2 . Third, policy revenue remains superficially close under the failed approximator because the continuation term is evaluated at the current state and often has little leverage over the action ranking. These results demonstrate that rerun reproducibility, predictive generalization, and policy validity are distinct properties.

1.1 Contributions

- An end-to-end numerical audit that distinguishes exact file regeneration from validity of the experimental comparison.
- A common-random-number re-evaluation of 25,500 mechanism-policy runs and 20,400 value-approximation policy runs.
- An instance-level holdout test showing that row-wise or in-sample fit can conceal complete failure to generalize across problem instances.
- A correction to state-error summaries that separates zero-capacity terminal states from decision-relevant states.
- A ten-stage minimum validation protocol that RM teams, procurement officers, and consultants can use to audit third-party algorithm claims before supporting investment or deployment decisions.

2. BACKGROUND AND RELATED WORK

2.1 Seat Protection and Dynamic Pricing

Littlewood's rule and Belobaba's EMSR extensions formalized the tradeoff between accepting lower-fare demand and preserving inventory for later higher-fare demand [1], [2]. Foundational surveys describe the development and scope of perishable-asset and transportation revenue management [5], [6]. Talluri and Van Ryzin place these controls within the broader theory of quantity- and price-based revenue management [3], including network bid-price controls [15]. Dynamic-pricing and airline seat-allocation models similarly value remaining inventory through finite-horizon opportunity costs [4], [16]. In both traditions, an inaccurate continuation value can produce premature inventory disposition or excessive protection.

2.2 Approximate Dynamic Programming

Exact stochastic dynamic programming is generally infeasible at airline scale because network, itinerary, fare-class, customer-choice, and time dimensions expand the state space. Approximate dynamic programming and reinforcement-learning methods replace the exact value function with parametric, nonparametric, or simulation-based approximations [7]–[9]. Recent airline and network-pricing studies illustrate the continued use of optimization, reinforcement learning, and learned decision policies in revenue-management settings [17]–[19]. A high predictive fit is not sufficient by itself: approximation quality must be measured out of sample and, more importantly, through induced decisions and policy regret. This distinction matters operationally because prediction error is costly only through the decisions it changes, while an apparently accurate model may still rank feasible pricing or availability actions incorrectly.

2.3 Simulation Design and Reproducibility

Variance-reduction methods such as common random numbers compare alternative systems under aligned exogenous shocks [10], [11]. Verification asks whether code implements the intended model; validation asks whether the model is adequate for its stated purpose [12]. Reproducible computation is a necessary supporting condition, but identical reruns do not validate model semantics, baseline fairness, or causal attribution [13], [14]. The present audit applies this distinction to a revenue-management case in which policy order, state timing, and holdout construction materially affect interpretation.

3. EXPERIMENTAL SETTING

3.1 Synthetic Carrier–Market Design

The experiment contains 17 synthetic carrier profiles: six network full-service, five regional full-service, four low-cost, and two hybrid profiles. Profiles vary in physical capacity, number of fare classes, base fares, demand intercepts, time coefficients, and price elasticities. Thirty synthetic markets are divided into ten high-, ten medium-, and ten low-demand multiplier ranges. Ten integer seeds are applied to every carrier–market combination, yielding 5,100 scenario identifiers. No profile represents an identified airline, and the parameter ranges are scenario-design assumptions rather than empirical estimates.

The booking horizon comprises 12 six-hour epochs spanning the final 72 hours. Let x_t denote remaining capacity, h_t the hours before departure, p_{tf} the multiplicative price factor for fare class f , and λ_{tf} the corresponding expected arrivals. Demand intensity is log-linear in base demand, normalized time, log price, market demand, and seasonality. Realized arrivals are sampled from Poisson distributions. Revenue is incremental revenue earned during the final 72-hour window; revenue from seats sold before the simulated window is excluded.

$$\log \lambda_{tf} = \alpha_f + \beta_f(72 - h_t)/72 - \epsilon_f \log(p_{tf}) + 0.3 \log(m) + 0.2s.$$

3.2 Policies

The EMSR-b-style benchmark holds prices at their base values and closes lower fare classes when remaining capacity is at or below a normal-approximation protection level computed from remaining higher-fare demand. Four heuristic variants search a discrete price-factor grid $\{0.85, 0.925, 1.00, 1.075, 1.15\}$:

- Original EWP: one-step projected demand multiplied by a time-varying realization factor.
- Availability only: original EWP plus closure of the lowest fare class below 30% remaining capacity.
- Aggregate look-ahead: a deterministic epoch-by-epoch future-demand aggregation in place of the one-step term.
- Combined look-ahead and availability: aggregate look-ahead plus the 30% closure rule.

The archive labels the third component “exact continuation.” That term is not retained here. The implementation substitutes expected arrivals for stochastic transitions, truncates aggregate expected sales to an integer next state, and does not integrate over Poisson outcomes. It is therefore a deterministic aggregate look-ahead or fluid-style approximation, not an exact stochastic Bellman value.

3.3 Original and Corrected Randomization

In the archived implementation, a random number generator is initialized once for each scenario and then consumed sequentially by policies. Because policies are evaluated in a fixed order, each receives a different demand stream, and policy comparisons are confounded by stochastic-stream misalignment.

For the corrected evaluation, we implement strict common random numbers through pre-generated latent uniform variates held fixed across policies. For each scenario, epoch, and fare class, a policy-independent variate U is drawn from $\text{Uniform}(0,1)$ and reused for every policy. Policy-specific Poisson arrivals are obtained by applying the inverse Poisson cumulative distribution function to U using that policy’s demand intensity. This aligns policies on the same latent shocks and makes the coupling invariant to policy evaluation order and to variable random-number consumption by direct Poisson samplers. The shared initial-capacity draw remains policy-independent as in the original design.

3.4 Value-Approximation and Holdout Design

Twenty-four small deterministic dynamic-programming instances are generated across capacities 4, 6, 8, and 10, with six instances per capacity and two fare classes. For each instance, values are recorded over 12 epochs and all capacity states, producing 2,304 state-value rows. A degree-2 polynomial regression uses ten engineered features: normalized remaining capacity, normalized time, log total demand, demand per remaining seat, normalized mean fare, fare spread, high-fare demand share, and three interaction terms.

The archived code comments describe an 18-instance training and 6-instance holdout. The executable split actually assigns the first four and last two instances at each capacity to training and holdout, respectively: 16 training instances and 8 held-out instances. This corrected split description is used

throughout. The split is performed by complete instance, not by row, preventing states from the same instance appearing in both sets.

3.5 Metrics and Statistical Inference

Policy outcomes include mean incremental revenue, revenue lift relative to the EMSR-b-style benchmark, scenario win rate, and paired revenue difference. Value-function evaluation uses training and holdout R^2 , MAE, maximum absolute error, and per-instance R^2 . We resampled the 510 carrier–market clusters with replacement for 4,000 replicates, retaining all ten seeds within each selected cluster. We report the 2.5th and 97.5th percentiles of the bootstrap distribution of the paired mean revenue difference. The bootstrap seed was fixed at 42. All policy comparisons use the same cluster bootstrap procedure unless otherwise noted.

4. REPRODUCIBILITY AUDIT

4.1 Archive and Mechanical Rerun

We distinguish the legacy archive being audited from the corrected V11 reproducibility release accompanying this article. The supplied archive contains three Python scripts, four principal CSV/JSON output groups, and a frozen high-level configuration. The unmodified mechanism script regenerates 25,500 policy rows and 1,008 state-error rows. Aggregate means match the archived values to displayed precision; byte-level hashes differ only because regenerated floating-point text differs at approximately 10^{-11} to 10^{-13} and because environment-specific serialization is not pinned. This establishes numerical rerun reproducibility under the original implementation, not experimental validity.

Table 1
Audit of the legacy pre-correction archive.

Audit item	Observed result	Interpretation
Unmodified 2×2 rerun	25,500 rows; displayed means reproduced	Pass: numerical rerun
State-error rerun	1,008 rows; displayed aggregate reproduced	Pass: numerical rerun
Byte-identical CSV	No; only negligible float-text differences	Needs environment/serialization pinning
README version	Identifies package as V7	Fail: stale metadata
Claimed V9 manuscript	Not included in archive	Fail: incomplete release
Main 51,000-run experiment	Not included	Fail: incomplete evidence chain
Environment lock file	Not included	Fail: dependency versions unspecified

4.2 Common-Random-Number Stress Test

Table 2 compares the original policy-order streams with the corrected CRN evaluation. The sign and broad ranking are stable, but the apparent advantage of aggregate look-ahead is smaller under CRN: its shortfall widens from – 1.31% to – 2.04%. The availability threshold remains harmful both alone and in combination.

Table 2
Revenue lift relative to EMSR-b before and after CRN correction

Policy	Original streams	CRN audit	Change (pp)
Original EWP	-15.53%	-15.84%	-0.31
Availability only	-17.52%	-18.62%	-1.10
Aggregate look-ahead	-1.31%	-2.04%	-0.73
Look-ahead + availability	-5.32%	-5.90%	-0.58

4.3 Post-Decision-State Audit

For a candidate action a_t , a continuation term should normally evaluate the state after current expected or realized sales and at the next epoch. A generic one-step action value is

$$Q(s_t, a_t) = r(s_t, a_t) + \gamma V_{t+1}(s_{t+1}).$$

The audited code instead evaluates the value approximation at the current time and current remaining capacity. The deterministic aggregate-look-ahead function also omits candidate price factors and post-decision remaining capacity; for a fixed state it therefore adds the same future-value constant to every candidate price. It cannot affect the candidate ranking except indirectly through replacement of the original price-dependent one-step term. Consequently, the 2×2 table does not identify the causal contribution of a correctly specified continuation value. It identifies the effect of substituting one scoring term with an action-invariant aggregate offset.

5. HOLDOUT VALIDATION RESULTS

5.1 Value-Function Generalization

The complete-instance holdout reveals severe generalization failure. The 16-instance training model reaches $R^2=0.966$ and training MAE of \$96.99. On the eight held-out instances, aggregate R^2 is – 1.937, MAE is \$253.06, and maximum absolute error is \$765.89. Every held-out instance has negative R^2 . The extreme per-instance values are not typographical errors: when within-instance reference-value variance is small, moderate absolute errors generate very negative R^2 . The economically relevant conclusion is that predictions are worse than an instance-specific mean benchmark across all held-out instances.

Table 3
Instance-level value-function validation

Metric	Training	Holdout
Instances	16	8
State-value rows	1,536	768
R ²	0.966	-1.937
MAE	\$96.99	\$253.06
Maximum absolute error	\$743.11	\$765.89

5.2 Policy-Level Evaluation Under CRN

Under the corrected CRN evaluation, the model fitted on all 24 instances yields mean revenue of \$21,250.83, or - 3.26% relative to EMSR-b. The model fitted on only the 16 training instances yields \$21,154.09, or - 3.70%. Both cluster-bootstrap intervals for mean revenue difference remain below zero. Scenario win rates are close to 50%, but a win-rate near one half does not offset the negative mean because losses are larger than gains in economically important scenarios.

Table 4
Corrected CRN policy evaluation over 5,100 scenarios

Policy	Mean revenue	Lift	Win rate	Cluster-bootstrap difference CI
EMSR-b-style benchmark	\$21,967.05	0.00%	—	—
Original EWP	\$18,488.33	-15.84%	15.3%	[-\$4,010, -\$2,980]
Distilled, all 24 instances	\$21,250.83	-3.26%	54.4%	[-\$1,087, -\$369]
Distilled, 16 training instances	\$21,154.09	-3.70%	54.5%	[-\$1,200, -\$449]

The two distilled variants generate exactly identical revenue in 78.7% of scenarios and identical seat counts in 97.0%. This high action-level similarity, despite the failed held-out value fit, indicates weak policy sensitivity to the learned continuation estimate. It should not be interpreted as evidence that value-function generalization is unnecessary. Rather, it motivates direct reporting of action agreement and policy regret on small instances where a valid stochastic benchmark can be computed.

5.3 Non-Degenerate State-Error Reporting

The archived error table contains 1,008 rows. Of these, 144 correspond to zero remaining capacity, for which both the reference value and heuristic estimate are zero. The implementation assigns these rows a percentage error of zero. Including them produces a mean error of - 47.8% and an “underestimation rate” of 85.7%. Excluding states in which the reference value is zero leaves 864 decision-relevant rows;

every one is underestimated, and mean percentage error is – 55.8%. The latter is the appropriate primary summary because percentage error is undefined when the denominator is zero.

Table 5
Effect of degenerate states on error summaries

Sample definition	Rows	Mean percentage error	Underestimation rate	MAE
All recorded states	1,008	–47.8%	85.7%	\$166.27
Reference value > 0	864	–55.8%	100.0%	\$193.99

6. DISCUSSION

6.1 Mechanical Reproducibility Is Not Validity

The strongest positive result of the audit is that the archived scripts do regenerate their principal tables. That achievement matters: it permits independent diagnosis and prevents an unsupported result from remaining opaque. Nevertheless, reproducibility at the level of deterministic rerun preserves implementation defects as faithfully as intended behavior. A credible validation chain must separately establish code verification, experimental identification, statistical uncertainty, and domain validity.

6.2 What the Negative Holdout Result Means

The holdout finding is not a reason to conceal or discard the study. It is the most informative scientific result. A compact polynomial fitted to pooled state rows learned correlations that did not transfer across newly generated fare and demand configurations. Future approximators should normalize monetary scale more carefully, enforce monotonicity in capacity and remaining time, incorporate instance-level random effects or richer structural features, and be assessed through repeated grouped cross-validation. Most importantly, model selection should use policy regret rather than training R^2 .

6.3 Availability Control

The tested rule closes only the lowest fare class when remaining inventory falls below 30% of capacity. Its performance is consistently worse than the corresponding no-closure version. This does not show that availability control is generally harmful; EMSR-b itself is an availability-control mechanism. It shows that an uncalibrated scalar threshold should not be treated as a substitute for nested protection levels or marginal-capacity values.

6.4 Implications for Airline Analytics Practice

Before shadow-mode testing, an airline or technology provider should require an immutable scenario manifest, policy-independent exogenous random inputs, a baseline unit-test suite, post-decision state checks, grouped holdout families, and distributional performance reporting. Production latency should be measured separately from simulation runtime and should include feature retrieval, model inference,

policy search, and integration overhead. The current experiment supports none of the claims necessary for live deployment; it supports a reproducibility and validation workflow.

For airlines evaluating third-party RM technology, these findings translate directly into procurement practice. A vendor demonstration that shows revenue improvement in a simulation should not be accepted at face value. Procurement teams should require vendors to provide: (1) the complete simulation code and raw data, not just summary tables; (2) evidence that all compared policies share identical exogenous demand streams; (3) a unit-tested baseline implementation with documented protection-level formulas; (4) holdout results on problem instances not used for training or tuning; and (5) distributional performance reporting, including worst-case scenarios and win rates, not just mean revenue. Vendors who cannot or will not provide these materials should be treated as presenting unverified claims, regardless of how impressive the headline numbers appear.

For internal RM analytics teams, the audit highlights the importance of model governance before deployment. A model that performs well in training may fail completely in new markets, demand regimes, or competitive environments. The holdout result in this study—training $R^2=0.966$ but instance-level holdout $R^2=-1.937$ —illustrates that excellent in-sample fit can coexist with complete out-of-sample failure. RM teams should therefore establish internal validation gates before any algorithmic pricing or inventory control is moved into production: (1) frozen test scenarios that are never used for training or tuning; (2) shadow-mode comparison against the existing production policy for at least one full booking season; (3) monitoring of action agreement, not just revenue, to detect when a new policy makes systematically different decisions; and (4) explicit rollback criteria if performance degrades beyond a predefined threshold.

6.5 Layered Validation Framework

The broader methodological contribution is a layered validation framework rather than a diagnosis limited to one implementation. The ten stages in Section 7 can be organized into four evidence layers. The semantic and implementation layer covers model specification, baseline verification, exogenous-scenario construction, and post-decision-state tests; it asks whether the experiment represents the intended decision problem. The statistical and generalization layer covers grouped validation, policy-level metrics, and paired inference; it asks whether the measured effect is stable beyond fitted states and individual random seeds. The computational and artifact layer covers runtime reporting and reproducible release; it asks whether an independent reviewer can reconstruct the evidence chain. The operational and external-validity layer asks whether offline findings survive real data, workflow constraints, and controlled exposure.

These layers are deliberately sequential. Passing a later layer cannot repair failure in an earlier one: a precisely estimated revenue difference is not meaningful when policies receive different exogenous demand streams, and a perfectly archived model is not operationally valid merely because its files can be rerun. Conversely, an early-layer pass is necessary but insufficient for deployment. This ordering turns a checklist into a decision rule: evidence should advance only when the preceding layer has been

independently reviewed and unresolved exceptions have been documented. The framework is derived from the present audit and therefore remains a proposed methodology; its transferability should be evaluated across other RM formulations, demand models, and operational settings.

A practical handoff package should mirror the four layers. It should include a scenario and model specification, a versioned policy implementation with baseline tests, an evidence bundle containing paired outcomes and holdout results, and an operational record identifying reviewers, residual limitations, approval thresholds, and rollback triggers. This structure helps a governance committee distinguish a software demonstration from a controlled policy experiment and makes responsibility for each validation claim explicit.

6.6 Applying the Framework to Live RM Systems

Application to a live RM system should proceed through staged exposure. First, an offline audit should freeze the incumbent policy, business objective, booking horizon, demand inputs, and economic metric before comparative results are viewed. Second, shadow-mode operation should test data latency, feature availability, action feasibility, and agreement with the incumbent without changing customer offers. Third, a limited pilot should predefine eligible markets, traffic allocation, monitoring intervals, economic guardrails, and stop conditions. Only after technical reliability and incremental commercial performance are separately demonstrated should wider deployment be considered. The transition between stages should be governed by prospective thresholds rather than favorable retrospective interpretation. This sequence does not convert synthetic evidence into field validity; it defines the additional evidence required before a simulation result can support a real pricing or inventory decision.

At the industry level, broader concerns about reproducibility and replicability in computational and simulation research are directly relevant to RM [21]–[24]. Conferences such as AGIFORS Revenue Management Study Group and INFORMS Revenue Management and Pricing Section could play a role in establishing community standards for simulation-based RM research, including required reporting of random-number handling, baseline verification methods, and holdout design. Journals could require reproducibility packages as a condition of acceptance, with editorial checks that the code actually regenerates the reported tables. The ten-stage protocol proposed in Section 7 is intended as a starting point for such community standards, not a definitive checklist. The goal is not to impose unnecessary burden on researchers, but to ensure that simulation evidence used in material airline technology decisions is methodologically sound.

7. MINIMUM VALIDATION PROTOCOL

Table 6
Proposed minimum protocol for simulation-based RM policy studies

Stage	Required evidence	Failure prevented
1. Model specification	State, action, transition, reward, timing, units	Semantic ambiguity
2. Baseline verification	Analytical cases and unit tests	Weak or incorrect comparator
3. Exogenous scenarios	Frozen latent uniforms or demand paths	Policy-order confounding
4. Post-decision tests	Inventory and time transition assertions	Action-invariant continuation values
5. Grouped validation	Hold out complete carriers, markets, or instances	Row-level leakage
6. Policy metrics	Regret, action agreement, tail loss, win magnitude	Misleading predictive fit
7. Statistical inference	Paired cluster intervals and seed sensitivity	Pseudo-replication
8. Computational reporting	Hardware, software, p50/p95/p99, timing scope	Unsupported latency claims
9. Artifact release	Hashes, environment lock, raw and derived data	Unverifiable results
10. External validity	Shadow mode or real-data replication	Synthetic-to-operational overreach

8. LIMITATIONS

This study is an audit of one synthetic implementation, not a general comparison of dynamic pricing and EMSR-b. The demand model omits customer choice among fare classes, cancellations, no-shows, group demand, network displacement, origin–destination control, competitor response, and learning. The 72-hour horizon excludes most of the booking curve. Carrier profiles and market multipliers are constructed rather than estimated from airline data.

The deterministic small-instance benchmark is not an exact stochastic dynamic program. A future benchmark should integrate over Poisson transition probabilities, enforce shared physical capacity across fare-class demand, and evaluate continuation at the next post-decision state.

The audit was conducted after inspecting archived results and was not preregistered. Bootstrap intervals quantify sampling variability over the fixed synthetic carrier–market design but do not establish population validity. Finally, the article does not claim that the tested heuristic, aggregate look-ahead, or distilled approximation is ready for airline deployment. Additionally, the cross-model portability of this

validation framework remains to be evaluated in independent revenue-management environments beyond the synthetic implementation audited here.

9. REPRODUCIBILITY AND DATA AVAILABILITY

The complete reproducibility package accompanies this submission as supplementary material. It contains the corrected CRN simulation scripts (`code/paper1_v10_strict_crn_simulation.py`), figure-generation code (`code/generate_figures.py`), bootstrap CI generator (`code/generate_bootstrap_ci.py`), frozen configuration (`data/simulation_config.json`), all raw CSV/JSON outputs, SHA-256 manifest (`data/SHA256_MANIFEST.json`), environment requirements (`requirements.txt`), and the manuscript itself. The corrected CRN analysis was produced using pre-generated policy-independent latent uniform variates held fixed across policies for every scenario, epoch, and fare class. Policy-specific Poisson arrivals were generated by applying the inverse Poisson cumulative distribution function to these shared variates under each policy's demand intensity. This construction makes the stochastic coupling independent of policy evaluation order and variable random-number consumption by direct Poisson samplers. The results reported in this article correspond to release v11.0. All results can be regenerated by running the strict-CRN simulation script followed by the bootstrap and figure scripts. A versioned public repository will be activated after its contents and access permissions have been independently checked.

10. CONCLUSION

This study demonstrates why a simulation result can be reproducible and still be methodologically invalid. The archived airline revenue-management experiment mechanically regenerates its tables, yet policy comparisons use different stochastic streams, the continuation term is evaluated at the wrong state, a deterministic look-ahead is called exact, zero-value states distort percentage-error summaries, and a high in-sample value-function fit collapses under complete-instance holdout. Correcting random-stream alignment changes reported policy effects and confirms that none of the tested heuristics outperforms the EMSR-b-style benchmark in mean revenue. The aggregate-look-ahead and distilled variants recover much of the original heuristic's loss, but their remaining shortfalls and failed value generalization preclude deployment claims.

The scientific contribution is therefore methodological and constructive: simulation-based RM research should freeze policy-independent exogenous scenarios, distinguish pre- and post-decision states, validate on complete unseen problem instances, report policy regret rather than training fit alone, exclude undefined percentage errors, and publish a complete executable artifact. These requirements turn negative findings into usable knowledge and reduce the risk that implementation artifacts are mistaken for revenue-management innovation.

Declarations

Author Contribution

Z.Z. conceived the study, designed the experiments, conducted the simulations, analyzed the data, and wrote the manuscript.

Acknowledgement

The author thanks the developers of the open-source scientific Python ecosystem (NumPy, SciPy, pandas, scikit-learn, matplotlib) that made this research possible.

Data Availability

The simulation code, configuration files, raw output data, and analysis scripts used in this study are available in the GitHub repository: <https://github.com/zhaormax-dotcom/askzhu>

References

1. Littlewood, K. 1972. Forecasting and control of passenger bookings. In Proceedings of the 12th AGIFORS Symposium, 95–117.
2. Belobaba, P. P. 1989. Application of a probabilistic decision model to airline seat inventory control. *Operations Research* 37(2):183–197. <https://doi.org/10.1287/opre.37.2.183>
3. Talluri, K. T., and G. J. van Ryzin. 2004. *The Theory and Practice of Revenue Management*. Boston: Springer. <https://doi.org/10.1007/b139000>
4. Gallego, G., and G. van Ryzin. 1994. Optimal dynamic pricing of inventories with stochastic demand over finite horizons. *Management Science* 40(8):999–1020. <https://doi.org/10.1287/mnsc.40.8.999>
5. Weatherford, L. R., and S. E. Bodily. 1992. A taxonomy and research overview of perishable-asset revenue management. *Operations Research* 40(5):831–844. <https://doi.org/10.1287/opre.40.5.831>
6. McGill, J. I., and G. J. van Ryzin. 1999. Revenue management: Research overview and prospects. *Transportation Science* 33(2):233–256. <https://doi.org/10.1287/trsc.33.2.233>
7. Powell, W. B. 2011. *Approximate Dynamic Programming: Solving the Curses of Dimensionality*. 2nd ed. Hoboken, NJ: Wiley. <https://doi.org/10.1002/9781118029176>
8. Bertsekas, D. P. 2017. *Dynamic Programming and Optimal Control*. 4th ed. Belmont, MA: Athena Scientific.
9. Sutton, R. S., and A. G. Barto. 2018. *Reinforcement Learning: An Introduction*. 2nd ed. Cambridge, MA: MIT Press.
10. Law, A. M. 2015. *Simulation Modeling and Analysis*. 5th ed. New York: McGraw-Hill.
11. Glasserman, P. 2003. *Monte Carlo Methods in Financial Engineering*. New York: Springer. <https://doi.org/10.1007/978-0-387-21617-1>

12. Kleijnen, J. P. C. 1995. Verification and validation of simulation models. *European Journal of Operational Research* 82(1):145–162. [https://doi.org/10.1016/0377-2217\(94\)00016-6](https://doi.org/10.1016/0377-2217(94)00016-6)
13. Ioannidis, J. P. A. 2005. Why most published research findings are false. *PLoS Medicine* 2(8):e124. <https://doi.org/10.1371/journal.pmed.0020124>
14. Peng, R. D. 2011. Reproducible research in computational science. *Science* 334(6060):1226–1227. <https://doi.org/10.1126/science.1213847>
15. Talluri, K. T., and G. J. van Ryzin. 1998. An analysis of bid-price controls for network revenue management. *Management Science* 44 (11, part 1): 1577–1593. <https://doi.org/10.1287/mnsc.44.11.1577>
16. Feng, Y., and B. Xiao. 2001. A dynamic airline seat inventory control model and its optimal policy. *Operations Research* 49(6):938–949. <https://doi.org/10.1287/opre.49.6.938.10026>
17. Gatti Pinheiro, G., M. Defoin-Platel, and J.-C. Régim. 2022. Outsmarting human design in airline revenue management. *Algorithms* 15(5):142. <https://doi.org/10.3390/a15050142>
18. Duijndam, K., G. Koole, and R. van der Mei. 2025. Bayesian reinforcement learning to optimize paid ancillary revenue in the airline industry. *Journal of Revenue and Pricing Management* 24:551–567. <https://doi.org/10.1057/s41272-025-00523-y>
19. Meng, H., N. Chen, and X. Gao. 2026. Reinforcement learning for continuous-time jump Markov decision processes with applications to network dynamic pricing. arXiv:2608.20680
20. Wang, K. K., M. D. Wittman, and T. Fiig. 2023. Dynamic offer creation for airline ancillaries using a Markov chain choice model. *Journal of Revenue and Pricing Management* 22:103–121. <https://doi.org/10.1057/s41272-022-00398-3>
21. Coveney, P. V., and R. R. Highfield. 2021. When we can trust computers (and when we can't). *Philosophical Transactions of the Royal Society A* 379(2197):20200067. <https://doi.org/10.1098/rsta.2020.0067>
22. van Kampen, A. H. C. et al. 2024. ENCORE: A practical implementation to improve reproducibility and transparency of computational research. *Nature Communications* 15:8117. <https://doi.org/10.1038/s41467-024-52446-8>
23. Lohmann, A., O. L. O. Astivia, T. P. Morris, and R. H. H. Groenwold. 2022. It's time! Ten reasons to start replicating simulation studies. *Frontiers in Epidemiology* 2:973470. <https://doi.org/10.3389/fepid.2022.973470>
24. Desai, A., M. Abdelhamid, and N. R. Padalkar. 2025. What is reproducibility in artificial intelligence and machine learning research? *AI Magazine* 46(2):e70004. <https://doi.org/10.1002/aaai.70004>

Figures

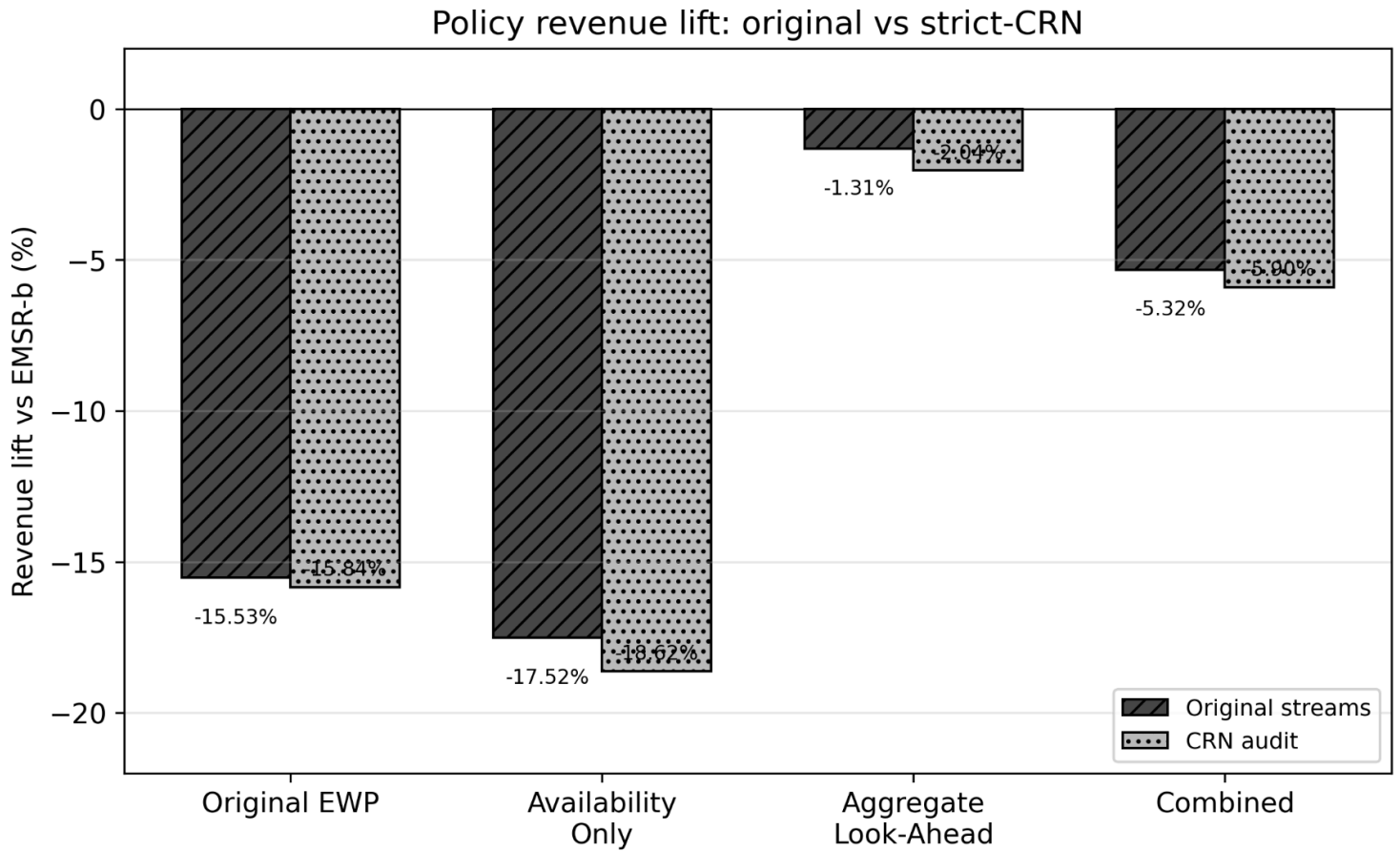


Figure 1

Sensitivity of policy conclusions to stochastic-stream alignment.

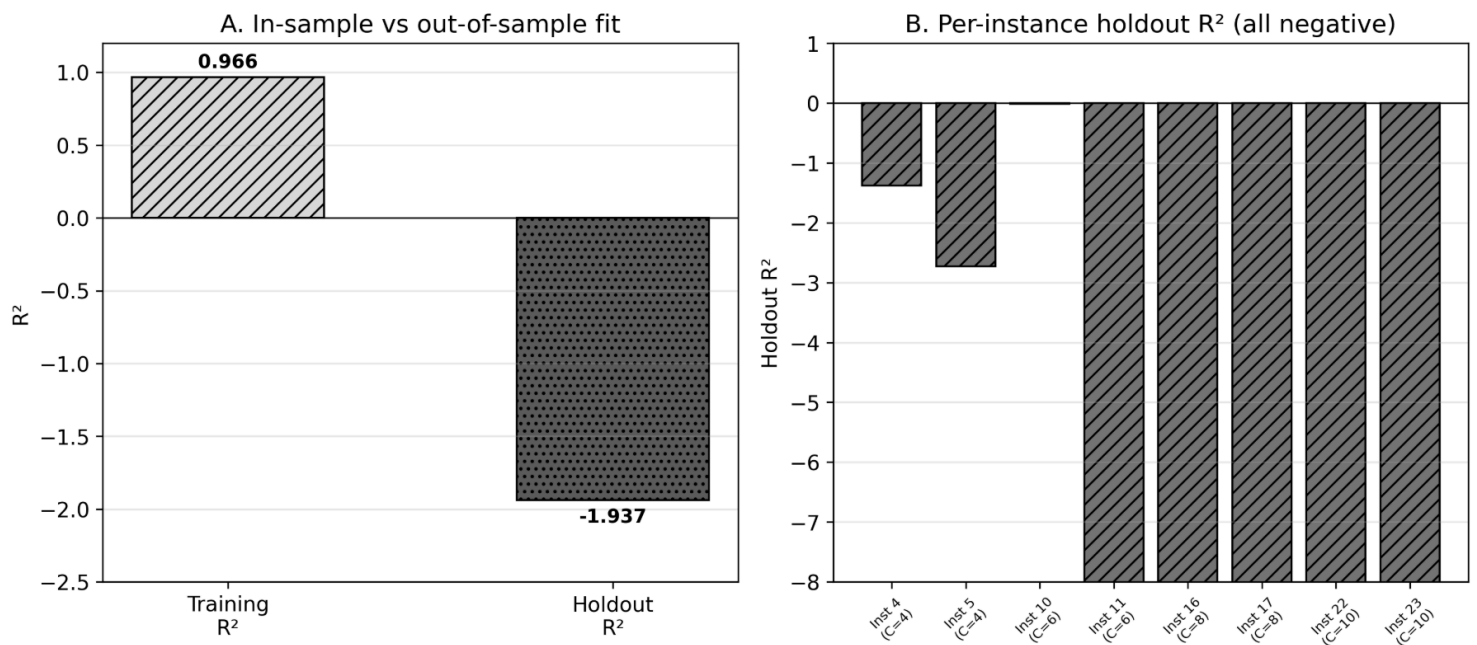


Figure 2

Training fit does not generalize to held-out instances.

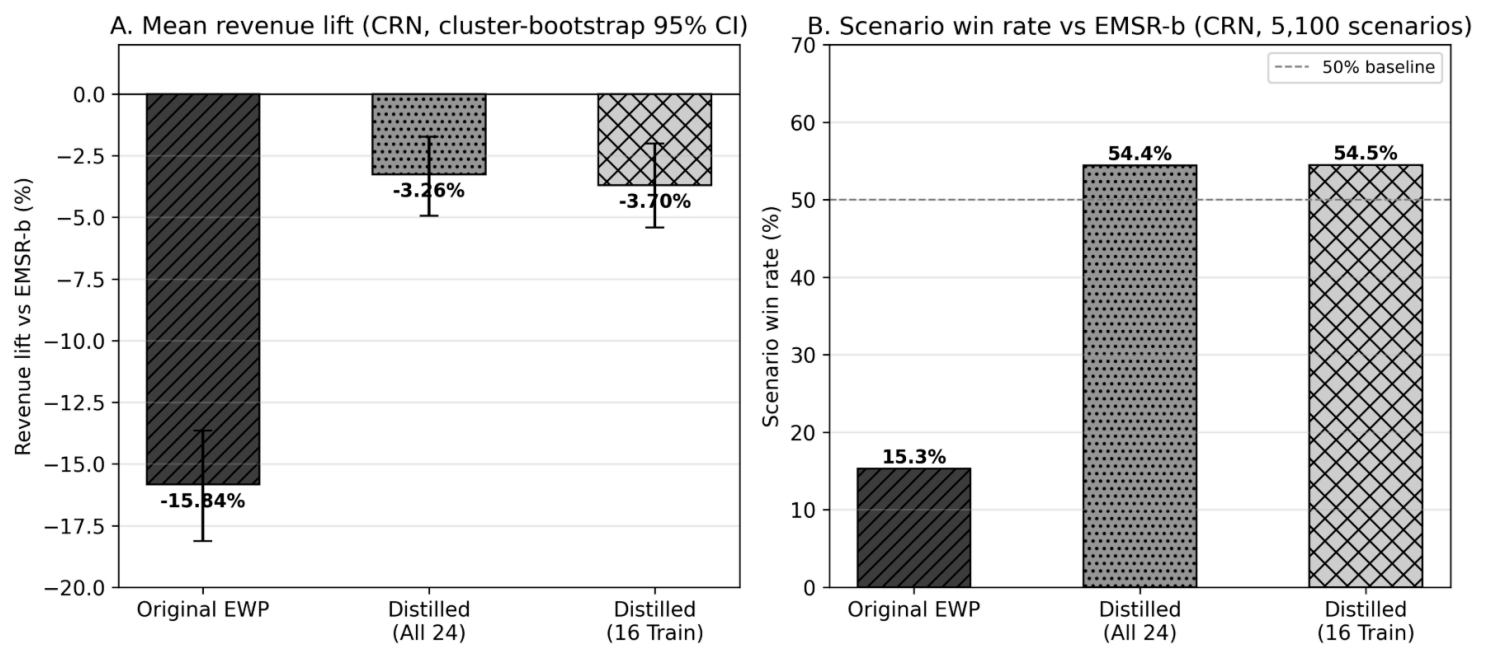


Figure 3

Corrected policy results show recovery but not superiority.