RETURNED-PREDICTOR AUDIT FOR EARLY MODEL SELECTION

Anonymous authors
Paper under double-blind review

ABSTRACT

Early model selection briefly trains several candidates, continues training on a subset, and uses validation to select a saved state for prediction. An improvement over random choices can come from favoring stronger candidates overall (usage), or assigning them to contexts where they help more (matching). We introduce Returned-Predictor Audit (RPA), an offline evaluation that separates these gains and traces them to the states actually returned. From complete records for the compared training plans, RPA evaluates random assignment at the policy's frequencies, computes the best reassignment at those frequencies using recorded test losses, and varies final validation while keeping early choices fixed. In an external two-task partial differential equation (PDE) study, a preselected pair yields the same states as the six-model pool in all 120 cases, with 41% less reconstructed training and validation work. In a six-task PDE study, positive usage and negative matching estimates nearly cancel each other out. On LCBench, pairing one fixed default with a second model chosen for each dataset reduces classification error by 0.250 percentage points compared to random assignment at the same frequencies. Broader final selection changes both measured matching and the opportunity left among returned predictors: matching can decrease while the policy's loss gap to the best reassignment under the new rule narrows.

1 INTRODUCTION

Selecting a model under a training budget often starts with short runs of several candidates. Early validation determines which candidates continue training, and final validation chooses a saved state for prediction (Li et al., 2018; Swersky et al., 2014; Pineda Arango et al., 2024). Such a policy is often judged by its final improvement over a baseline, but that number mixes two decisions: the first allocates training, and the second determines which trained state is used. A changed shortlist can therefore change the cost without changing the predictor, and a strong score can come from choices that a cheaper fixed procedure would also make.

Model portfolios and learned finetuning preferences supply evidence from previous tasks (Feurer et al., 2022; Pineda Arango et al., 2024). The policy we audit keeps a model chosen in advance, the *default*, and uses early validation to select a second model, its *companion*. Compared with a random companion, it can improve by choosing stronger candidates more often, which we call *usage*, or by assigning those candidates to datasets where they help more, which we call *matching*. Drawing companions at random with the policy's overall selection probabilities keeps usage and removes matching; each dataset still applies its own final validation.

Two findings from prior work motivate examining these decisions separately. Which models are in the pool affects transferability rankings (Nguyen et al., 2020; You et al., 2021; Agostinelli et al., 2022), and a fixed ranking can outperform task-aware metrics when a few models dominate a benchmark (Singh et al., 2026). These scores can reward model preference. Shortlist regret introduces a different issue by using the best test accuracy in the shortlist (Renggli et al., 2022, Section 3), whereas validation may return another member.

We introduce Returned-Predictor Audit (RPA) to evaluate the training plans and final selection together. A training plan is a *route*; Figure 1 shows five routes, each pairing the default with a different companion, that return only two distinct states on one dataset. Given complete outcomes for the declared routes, RPA compares the policy with independent assignment at its frequencies and

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

with the best reassignment using all recorded test losses. Changing final validation changes this entire comparison: it alters the states returned by the policy, its control, and the best assignment. Complete rows matter because they reveal how alternatives perform on the same dataset, a dependence that one observed outcome per instance can leave unidentified (Kallus, 2022). We then hold the early choices fixed and vary the saved states available to final validation.

We apply RPA to the pretrained PDE foundation models MPP, Poseidon, and MORPH (McCabe et al., 2024; Herde et al., 2024; Rautela et al., 2025) and to LCBench (Zimmer et al., 2021). Judged by total gain, adaptive companion selection improves on a random companion by 1.62 percentage points (pp) in the external PDE study. RPA attributes 1.41 pp to candidate usage and identifies a fixed pair that returns the full pool's states with 41% less reconstructed work. In a six-task PDE study, a near-zero total hides positive usage and negative matching; on LCBench, matching supplies about three quarters of the total gain, and extending final validation to all reached states lowers measured matching yet narrows the policy's loss gap to that rule's best reassignment. The final selector changes both the prediction and the benchmark for evaluating early assignment. The contributions are:

- **Auditing the Returned Predictor.** We trace each training plan to the state final validation returns, which identifies plans that yield the same predictor and lets the final selection rule change while early choices stay fixed.
- **Assignment Gain and Opportunity.** We jointly evaluate realized matching, the best same-frequency reassignment, and their loss gap under each final return rule.
- **Evidence in Two Domains.** PDE adaptation identifies cheaper routes to identical predictors, while classification reveals useful instance matching beyond candidate preference.

2 Returned-Predictor Audit

RPA evaluates a policy on a fixed menu of possible training plans and a fixed rule for choosing the saved state. The returned states determine prediction loss, while the executed plans determine cost.

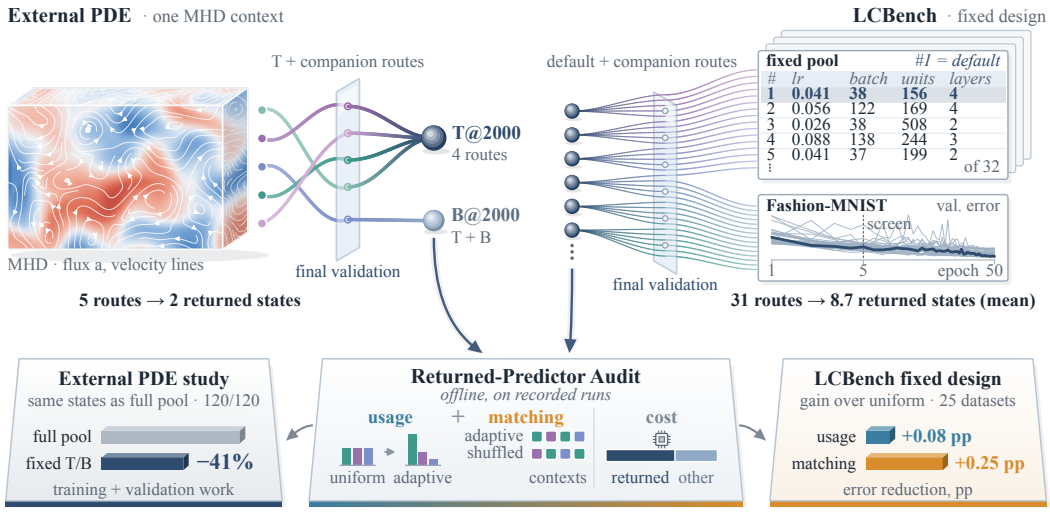


Figure 1: Audit overview. The magnetohydrodynamics (MHD) example uses cross-initialization selection; the field comes from the same generator at a calibration seed, while the route curves and audit miniatures are schematic. Shuffled denotes the frequency control. The 41% saving also holds when each run selects and evaluates its own state; costs combine training and calibrated validation. The 8.7 value is the LCBench mean returned-state count.

2.1 The Decision Being Evaluated

A *candidate* is a model with a specified training recipe. A *context* c identifies the dataset and fixed evaluation conditions, such as the initialization.

Within that context, a *route* k specifies a training plan, including which candidates continue and which stop. A *return rule* r uses validation to choose one saved state from those the route has computed. In Figure 1, five choices of companion return either Poseidon-T or Poseidon-B at update 2000.

Write a^r_{ck} for the selected state and L^r_{ck} for its test loss. The route cost C_{ck} charges all candidate training within the declared measurement protocol, including candidates not returned. Exact state identity records the candidate, update, and parameter state (Appendix A).

The early policy chooses route k with probability π_{ck} using its available observations. RPA requires the returned outcome of every compared route in each context. Candidate trajectories can be reused when pruning preserves each survivor's training trajectory, including its data order and learning-rate schedule (Appendix A). Three return rules illustrate the permitted-state sets: *cap-only* searches final survivors at the cap, *survivor-best* searches all their saved states, and *all-reached* searches every computed state. Final validation precedes evaluation of the selected state's test loss.

2.2 Separating Usage from Matching

A *frequency group* is a set of contexts over which route-use probabilities are held fixed; each PDE task is a group, and the 25 LCBench audit datasets form one group. Within one group, context weights w_c are positive and sum to one. Fix the return rule and write L_{ck} for the returned test loss; the policy selects route k with overall probability $q_k = \sum_c w_c \pi_{ck}$. The *uniform control* chooses every route with equal probability u_k. The *frequency control* draws route k with probability q_k independently for each context, so its expected route frequencies match the policy's while dataset-specific assignment is removed. Each control uses that context's own trained states and the same final validation rule. The two control losses and the adaptive loss are

$$\bar{L}^u = \sum_{c,k} w_c u_k L_{ck}, \qquad \bar{L}^q = \sum_{c,k} w_c q_k L_{ck}, \qquad \bar{L}^\pi = \sum_{c,k} w_c \pi_{ck} L_{ck}.$$

This gives the decomposition at the center of the audit:

$$\underbrace{\bar{L}^u - \bar{L}^\pi}_{\text{total gain}} = \underbrace{\bar{L}^u - \bar{L}^q}_{\text{usage gain}} + \underbrace{\bar{L}^q - \bar{L}^\pi}_{M(q,\pi)\,:\,\text{matching gain}}. \tag{1}$$

Usage measures the benefit of favoring some routes more often; matching measures the additional benefit of assigning those routes to particular contexts. With two choices and outcome defined as negative loss, the matching comparison is the finite-record form of the population average prescriptive effect (PAPE) (Imai & Li, 2023, Section 2.2). RPA extends this comparison to multiple training plans and evaluates the states that final validation returns. Given the complete records and fitted frequencies, it computes each context's control expectation $\sum_k q_k L_{ck}$ directly.

2.3 Matching Opportunity and the Report

An early policy can have weak matching even when other assignments would improve prediction. To measure the room for improvement at the same route frequencies, let $\Pi(w, q)$ contain the nonnegative assignment weights γ_{ck} whose context totals are w_c and route totals are q_k. The *matching opportunity* is the largest gain over the frequency control when all recorded test losses are available:

$$U(q) = \langle w q^\top, L \rangle - \min_{\gamma \in \Pi(w,q)} \langle \gamma, L \rangle. \tag{2}$$

The difference $U(q) - M$ is the gap between the adaptive policy's mean test loss and this return rule's test-informed optimum. Section 6 relates the opportunity to early information and resources.

Algorithm 1 computes each group's gains and bound, then takes their weighted average. The report also counts distinct returned states and gives the losses and costs of training fixed candidates directly.

3 Studies and Evaluation

We study PDE adaptation and classification learning curves under the settings in Table 1. In each study, the adaptive policy retains a default and uses early validation to choose a companion; direct models, fixed pairs, unrestricted shortlists, and staged pruning provide alternatives.

Algorithm 1 Returned-Predictor Audit with Return Rule r

Require: Complete outcomes for a feasible route menu, policy probabilities π_{ck}, context weights, and frequency groups.

Ensure: Returned-state counts, total/usage/matching gains, opportunity U, and adaptive/direct loss–cost comparisons.

1: Reconstruct each returned state a^r_{ck}, its loss L^r_{ck}, and route cost C_{ck}.
2: **for** each frequency group, using its normalized context weights **do**
3: Set $q_k = \sum_c w_c \pi_{ck}$ and evaluate $\bar{L}^u$, $\bar{L}^q$, and $\bar{L}^\pi$.
4: Compute the total, usage, and matching gains in Equation 1.
5: Solve the fixed-frequency allocation in Equation 2 for $U(q)$.
6: **end for**
7: Aggregate the group results; report returned-state counts, adaptive cost allocation, and direct-control losses and costs.

The external PDE study tests a previously specified default on tasks excluded from its construction. The six-task PDE study uses new instances of the same families that informed the default, whereas LCBench uses disjoint datasets to construct the defaults and evaluate selection.

Table 1: Settings for the main reports. PDE report costs sum four selection runs; the held-out run supplies independent evaluation. Same-run comparisons select and evaluate within one run. LCBench uses the evaluated run's validation and recorded time. Final validation returns the best saved state of the survivors; nRMSE is normalized root mean squared error.

	Six-task PDE	External PDE	LCBench
Contexts (clusters)	480 (48)	120 (24)	25 (25)
Candidates / companion routes	6 / 5	6 / 5	32 / 31
Screen / cap	100 / 3000 updates	100 / 3000 updates	5 / 50 epochs
Selection information	Cross-initialization	Cross-initialization	Own validation
Test loss	Relative nRMSE gap	Relative nRMSE gap	Classification error
Reported cost	Selection-run GPU-s	Selection-run GPU-s	Recorded seconds

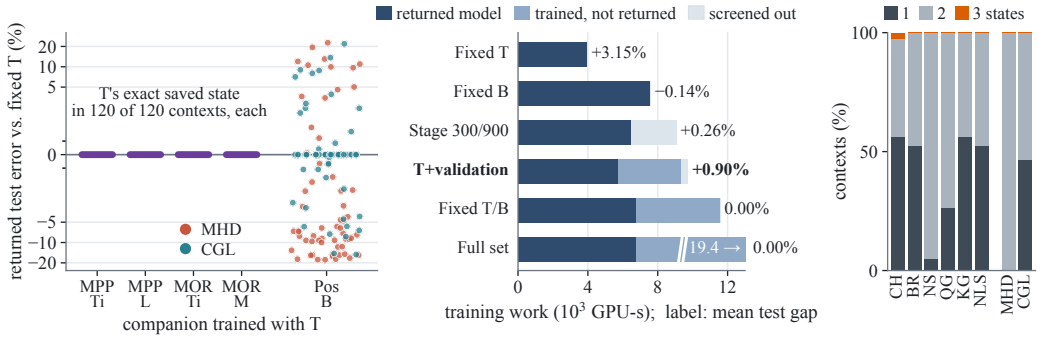
The PDE candidates are MPP-Tiny/Large (MPP-Ti/L), Poseidon-T/B (T/B), and MORPH-Tiny/Medium (MORPH-Ti/M), abbreviated Pos and MOR in figures (McCabe et al., 2024; Herde et al., 2024; Rautela et al., 2025). We specified T and T/B using adaptation evidence from the same PDE families before the six-task evaluation; external magnetohydrodynamics (MHD) and complex Ginzburg–Landau (CGL) did not enter that choice. T+validation denotes the policy that retains T and chooses a companion by early validation. Unrestricted Top-k instead retains the k early leaders, and the staged policies use successive halving (Jamieson & Talwalkar, 2016).

LCBench records 2,000 configurations on 35 datasets (Zimmer et al., 2021). We use ten *donor datasets* to construct the candidate pool and default, and 25 *audit datasets* to evaluate selection. PDE loss is $100(Y_c/D_c - 1)$, where Y_c is returned test nRMSE and D_c is the full-pool validation-selected reference error in that context (Takamoto et al., 2022; Koehler et al., 2024). Classification uses test error. Frequencies stay fixed within each PDE, or across the LCBench audit datasets: the former tests assignment within a family, the latter across datasets.

PDE intervals resample dataset seeds within each equally weighted task, carrying shared initialization and supervision observations together. LCBench resamples whole datasets; both procedures refit frequencies in every draw. Table 3 gives coverage levels. The 41% comparison uses same-run work including calibrated validation; the cross-initialization reports charge four selection runs and exclude the independently evaluated fifth run. Appendices G and C.4 give cost scopes and analysis roles.

4 Different Training Choices Can Return the Same Predictor

On the external PDE study, four of the five companion routes return T's exact saved state despite training different companions. Final validation ranks T ahead of every non-Poseidon candidate in every context, so only T/B can return a different predictor. Figure 2 links these repeated returns to the training paid for each choice.



(a) Five pairs, two predictors (b) Where the training goes (c) States per task

Figure 2: PDE returns and selection-run work. (a) Pair gap relative to T, on a symmetric-log axis linear within $\pm 5\%$. (b) Training by candidate status; labels give mean full-set-reference gap and the broken full-set bar reaches 19.4k GPU-s. (c) Distinct returns across companion routes.

The two-model T/B pair was specified before the six-task results and then used on the external tasks (Appendix D.2). It returns the same state as the full six-model pool in all 120 external contexts under both selection rules. Selecting and evaluating within each run, the pair uses 3.05k reconstructed device-seconds versus 5.17k for the full pool, including calibrated validation: 41% less work for the same returned states (Table 36).

Reconstruction adds measured training to the required validation calls, using initialization-state calibration; Appendix G gives the measured operation and sensitivity checks.

The adaptive policy spends 59% of its training on the candidate ultimately returned; most remaining work trains the other survivor. Direct B has a lower observed mean gap with 22% less training (Table 2b), and including calibrated validation preserves a similar saving (Table 36).

4.1 When Recipe Changes Alter the Returned Predictor

Changing the MPP objective and learning rate alters early scores, yet the full-pool reference returns the same state in all 80 paired external contexts across five recipes. Most changes in companion selection also leave the returned state unchanged. The intervention thus distinguishes score changes from changes in access to the predictor eventually selected.

Stage 100/600 transmits changes through competition for survival slots: an altered early score can displace another model and change the final predictor, even when the full pool never returns that candidate. Appendix E compares these transitions with the stable returns of Stage 300/900.

5 Total Gains Mix Candidate Usage and Instance Matching

A route that returns a different state creates a predictive alternative; the frequency control tests whether assigning that alternative to a particular context improves loss. Table 2 combines this comparison with the mean number of distinct returned states per context. The matching and opportunity columns use the same frequencies within each group.

In the six-task PDE study, the near-zero total combines a positive usage estimate with negative matching: the policy favors a useful mix of candidates, but independent draws at the same within-task frequencies assign that mix more effectively (Table 3). Matching is negative in five of six tasks, with pooled 95% interval $[-1.10, -0.02]$ pp; Table 6 reports the 97.5% sensitivity.

The external PDE gain comes mainly from usage, $+1.41$ of $+1.62$ pp, so a random mix at the policy's own frequencies already captures most of it. The LCBench matching estimate accounts for about three quarters of the total gain, with a positive fixed-design interval (Table 3).

Table 2: RPA report under survivor-best return. (a) Mean distinct returned states per context and gains at the declared route frequencies. (b) Prediction and cost. PDE units are relative-gap percent/pp and four-selection-run GPU-s; LCBench uses error percent/pp and recorded seconds. Returned share is the adaptive cost assigned to the returned candidate. Intervals appear in Table 3.

(a) Returned states and gain components

Study	States	Total	Usage	Matching	$U(q)$
Six-task PDE	1.59	-0.004	$+0.592$	-0.596	2.779
External PDE	1.77	$+1.621$	$+1.409$	$+0.212$	2.635
LCBench	8.68	$+0.328$	$+0.078$	$+0.250$	0.799

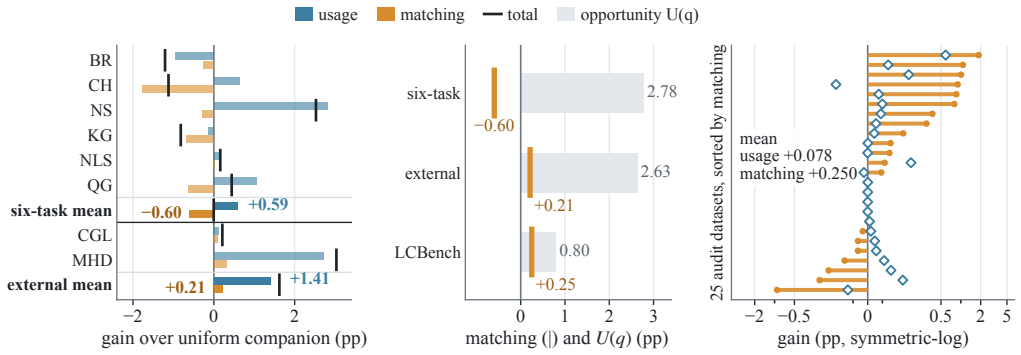
(b) Adaptive and direct alternatives

Study	Adaptive policy			Direct training		
	Loss	Cost	Returned (%)	Choice	Loss	Cost
Six-task PDE	1.41	8,476	54.3	Fixed T	1.69	3,854
				Fixed B	3.36	7,348
External PDE	0.90	9,731	58.9	Fixed T	3.15	3,973
				Fixed B	-0.14	7,599
LCBench	18.89	2,023	24.9	Donor pair	18.95	882

Table 3: Gain components with paired intervals (pp). PDE rows use pointwise 95% seed-cluster intervals; LCBench rows retain the two predeclared 97.5% fixed-design comparisons. Frequencies are refitted within each resample. Positive matching favors the adaptive assignment.

Study	Component	Gain	Interval	Coverage
Six-task PDE	Usage	$+0.592$	$[-0.05, 1.14]$	95%
	Matching	-0.596	$[-1.10, -0.02]$	95%
External PDE	Usage	$+1.409$	$[0.58, 2.31]$	95%
	Matching	$+0.212$	$[-0.17, 0.58]$	95%
LCBench	Total	0.328	$[0.062, 0.630]$	97.5%
	Matching	0.250	$[0.029, 0.499]$	97.5%

Figure 3a,c shows the task and dataset variation behind these means. The LCBench interval evaluates the donor-built pool and default as deployed; treating their construction as random as well, 2,000 donor/audit rebuilds give $[-0.017, 1.303]$ pp, and Appendix F.4 reports 3,765 further settings.

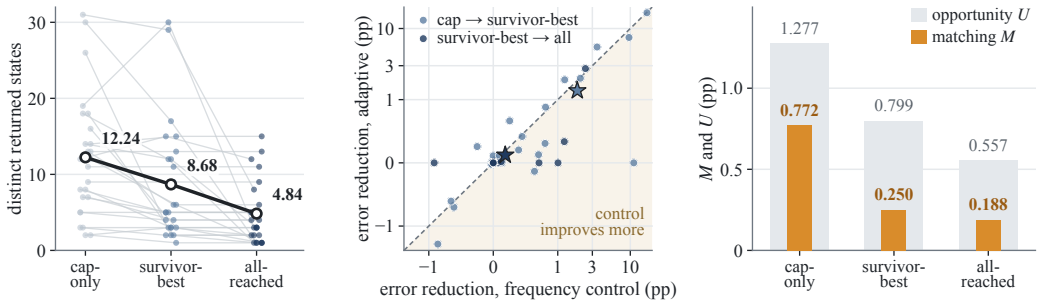


(a) PDE studies, by task (b) Realized versus available (c) LCBench fixed design

Figure 3: Gain sources and matching opportunity. (a) PDE task and equal-task usage/matching gains; ticks mark their sum. (b) Realized matching and test-informed $U(q)$ by study. (c) LCBench dataset matching, ordered by its value; diamonds show usage and annotations give means. PDE axes use relative-nRMSE-gap pp, LCBench uses classification-error pp.

5.1 The Final Return Changes the Comparison

We hold LCBench's early companion choices, route frequencies, and executed training fixed, then vary the states eligible for final validation. The resulting loss matrix changes for the adaptive policy, the frequency control, and the best reassignment.



(a) Returned-state counts (b) Paired error reductions (c) Matching and opportunity

Figure 4: Fixed epoch-5 choices under three return rules. (a) Returned-state counts. (b) Paired error reductions: control horizontal, adaptive vertical; stars are means. (c) Mean matching M inside opportunity U; the gray excess is $U - M$. Panel (b) uses symmetric-log axes, linear within ± 1 pp. Paired uncertainty is in Appendix F.

Searching earlier survivor states improves both workflows, with a larger observed error reduction for the control (Figure 4b). The resulting matching change is -0.522 pp, with paired 95% interval $[-1.467, 0.105]$ pp (Appendix F).

Table 4: Return-rule comparison on 25 LCBench datasets with early choices, frequencies, and training fixed. Adaptive and test-informed optimal errors are percentages; matching M, opportunity U, and loss gap $U - M$ are pp. Each optimum uses that row's return rule. Values are rounded independently; paired uncertainty for changes is in Appendix F.

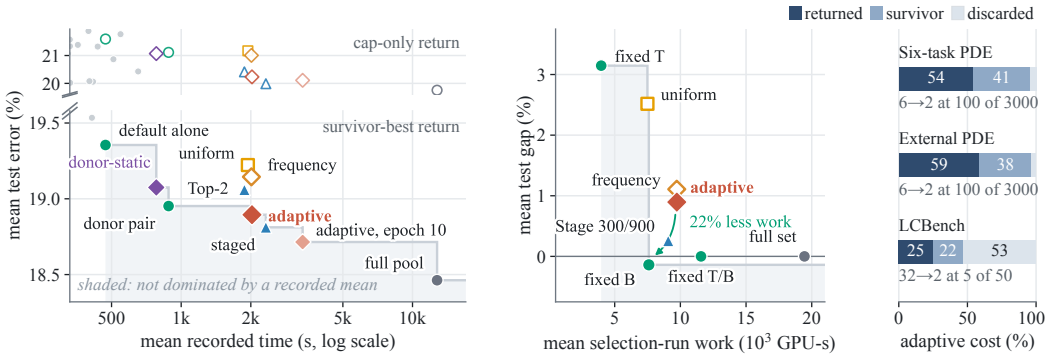
Return rule	Adaptive (%)	Optimum (%)	M	U	$U - M$
Cap-only	20.233	19.727	0.772	1.277	0.505
Survivor-best	18.895	18.346	0.250	0.799	0.549
All-reached	18.771	18.402	0.188	0.557	0.369

Table 4 and Figure 4c distinguish the two transitions. From cap-only to survivor-best, matching and opportunity both decrease, while the loss gap $U - M$ rises slightly. From survivor-best to all-reached, opportunity decreases more than matching, narrowing the gap from 0.549 to 0.369 pp. Here adaptive error falls while the optimal reassignment's error rises: final validation changes both sides of the comparison. A smaller matching gain can therefore accompany a smaller loss gap to the optimum under the new rule.

Under all-reached return, each route compares its survivor-best state with the same validation-best state from the shared prefix. This can merge returned states but cannot increase their number. The mean count falls from 8.68 to 4.84; Appendix B gives the formal argument.

6 Interpreting Matching Opportunity and Cost

The matching opportunity depends on the test losses of the states actually returned. We now interpret the bound in Equation 2 and ask how early observations and resource constraints limit the gains a policy can obtain. The statements below concern the routes the policy uses with positive frequency.



(a) LCBench fixed design (b) External PDE study (c) Cost by fate

Figure 5: Prediction and resource use. (a) LCBench: filled markers are survivor-best, open markers cap-only. (b) External PDE four-selection-run training costs, distinct from the same-run training-plus-validation headline. Shading marks nondominated plotted means. (c) Cost assigned to returned, other retained, and discarded candidates; source-specific units are retained.

6.1 When Returned-Loss Differences Permit Matching

The bound is a transport problem with fixed row and column totals (Peyré & Cuturi, 2019), which gives the following characterization.

Proposition 1 (Matching at fixed route frequencies)**.** *For a finite complete route menu and positive context and route marginals, every assignment with frequencies q satisfies $M(q,\pi) \le U(q)$. An assignment with access to all returned losses attains $U(q)$. Moreover,*

$$U(q) = 0 \quad \Longleftrightarrow \quad L_{ck} = \alpha_c + \beta_k \quad \textit{on the support.} \tag{3}$$

When the frequencies are fixed, context difficulty and a route's average advantage contribute the same loss under every assignment. Matching can help only when the advantage of one route over another differs across contexts. Moving a small amount of assignment weight between two contexts and two routes keeps both margins fixed. A nonzero change in loss allows an improving direction; if every such exchange has zero cost, the additive form in Equation 3 follows. Appendix B gives the proof and its constrained extensions.

Figure 3b compares matching with the test-informed opportunity. The six-task study has $U = 2.78$ pp and negative observed matching, distinguishing unfavorable assignment from an absence of useful differences. Table 4 shows why this benchmark must also follow the return rule: final validation changes the relative losses available for reassignment.

An early policy sees observations X. For a finite signal, applying the allocation problem to $\mathbb{E}[L_k \mid X = x]$ gives the opportunity exposed by that signal; the policy determines how much it realizes. The empirical $U - M$ can include both unobserved and unused relative-loss information. Appendix B.3 illustrates why complete outcome rows reveal more than separate route averages.

6.2 Comparing Adaptive Selection with Direct Training

Preserving route frequencies does not necessarily preserve cost: the same route can take different time on different datasets. For executed cost C_{ck} and a mean-cost budget, the constrained optimization evaluates losses together with costs through $L_{ck} + \lambda C_{ck}$, where $\lambda \ge 0$ is the budget multiplier (Appendix B). The direct and donor-trained policies below are alternatives that can actually be run.

The cost view also identifies a cheaper LCBench operating point: the donor fixed pair uses 56% less recorded time than adaptive selection, and the adaptive error minus the fixed pair's error is -0.057 pp with paired 95% interval $[-0.442, 0.365]$ pp (Figure 5a). A distribution over fixed training choices, fitted on donor validation with the adaptive policy's mean donor time as budget, gives another lower-cost alternative on the audit datasets (Appendix F). Comparing these complete procedures prices

the benefit of selection against direct training, consistent with the need for competitive baselines in scientific-model evaluation (McGreivy & Hakim, 2024).

LCBench spends 52.8% of recorded time on candidates discarded after screening, compared with 4.4% and 3.6% in the PDE studies, where continuing the second survivor costs more.

7 Relation to Prior Work

Selecting Pretrained Models and Checkpoints. Transferability scores rank pretrained representations for a target dataset: LEEP addresses classification, while LogME and regression-specific scores cover other targets (Nguyen et al., 2020; You et al., 2021; Nguyen et al., 2023). Pool composition affects these rankings (Bolya et al., 2021; Agostinelli et al., 2022), and checkpoint selection can ask how well a state will adapt further (Munn & Wei, 2025). The shortlist metric of Renggli et al. (2022) scores the member with the best test accuracy. RPA instead evaluates the state that validation actually returns after training.

Learning Curves and Resource Allocation. Hyperband, BOHB, and ASHA allocate training from partial evaluations (Li et al., 2018; Falkner et al., 2018; Li et al., 2020); PASHA also adjusts the maximum budget (Bohdal et al., 2023). Extrapolation, freeze–thaw methods, and curve priors guide continuation (Domhan et al., 2015; Swersky et al., 2014; Adriaensen et al., 2023), often using historical evidence (Mallik et al., 2023; Pineda Arango et al., 2024). In-context freeze–thaw BO selects an incumbent from all observed partial evaluations (Rakotoarison et al., 2024, Algorithm 1). These allocation and incumbent rules specify auditable routes and eligible states. Fixed-epoch discarding provides a simpler resource alternative (Egele et al., 2024).

Benchmarks and Selection Bias. ASlib and Syne Tune support evaluation across instances and budgets (Bischl et al., 2016; Salinas et al., 2022), with HPOBench, YAHPO Gym, and LCDB providing further benchmark settings (Eggensperger et al., 2021; Pfisterer et al., 2022; Mohr et al., 2023). The virtual best solver chooses the lowest-loss algorithm per instance without preserving its use frequency (Kerschke et al., 2019); $U(q)$ retains that constraint. BBC-CV corrects the optimism of the winning score by resampling predictions (Tsamardinos et al., 2018), while Post-Hoc Reversal studies changed rankings after further processing (Ranjan et al., 2024). Our controlled return-rule comparison measures a different quantity: the change in the gain over a fixed-frequency control.

Treatment-Rule Evaluation and Optimal Assignment. PAPE compares a personalized treatment rule with random treatment at the same overall proportion (Imai & Li, 2023); rank-weighted average treatment effects evaluate prioritization across treatment fractions (Yadlowsky et al., 2025). Observing one treatment outcome per unit can leave the joint individual outcomes partly unidentified (Kallus, 2022). RPA uses complete returned-loss rows to calculate the best assignment at fixed frequencies. Resource-aware decisions also have close precedents: capacity-constrained treatment assignment uses transport (Sunada & Izumi, 2025), and RouteLLM compares inference routing with cost-controlled random choices using paired response preferences (Ong et al., 2025). In RPA, each choice trains several states before validation selects one, so changing that final selection and charging all executed training exposes effects beyond the assignment alone.

8 Conclusion

RPA connects candidate preference and instance assignment to the predictions and costs they produce. Final validation changes the policy's outputs, its random control, and the same-frequency optimum, so the loss gap to that optimum can shrink even when the matching gain decreases. Comparisons of early model selection should therefore report the final selection rule and the resource measure alongside the gain, and evaluate them on records that cover the compared training plans.

Reproducibility statement

The supplement provides standardized candidate-state records for the three audit applications and CPU reconstruction of the reports and covered comparisons, with checks against printed values. Appendix A documents the interface and coverage. LCBench provides public classification records; our PDE data follow the generators in Appendix H. Pretrained weights are available from the model authors. Upon acceptance, we will release the finetuning code and processed records of the source training runs from which the standardized audit tables were built. Proofs, analysis roles, and complete supporting comparisons are given in Appendices B, C, and G.

References

Steven Adriaensen, Herilalaina Rakotoarison, Samuel Müller, and Frank Hutter. Efficient bayesian learning curve extrapolation using prior-data fitted networks. In *Advances in Neural Information Processing Systems*, 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/3f1a5e8bfcc3005724d246abe454c1e5-Abstract-Conference.html.

Andrea Agostinelli, Michal Pándy, Jasper Uijlings, Thomas Mensink, and Vittorio Ferrari. How stable are transferability metrics evaluations? In *Computer Vision – ECCV 2022*, volume 13694 of *Lecture Notes in Computer Science*, pp. 303–321. Springer, Cham, 2022. doi: 10.1007/978-3-031-19830-4_18. URL https://link.springer.com/chapter/10.1007/978-3-031-19830-4_18.

Bernd Bischl, Pascal Kerschke, Lars Kotthoff, Marius Lindauer, Yuri Malitsky, Alexandre Fréchette, Holger Hoos, Frank Hutter, Kevin Leyton-Brown, Kevin Tierney, and Joaquin Vanschoren. ASlib: A benchmark library for algorithm selection. *Artificial Intelligence*, 237:41–58, 2016. doi: 10.1016/j.artint.2016.04.003. URL https://www.sciencedirect.com/science/article/pii/S0004370216300388.

Ondrej Bohdal, Lukas Balles, Martin Wistuba, Beyza Ermis, Cedric Archambeau, and Giovanni Zappella. PASHA: Efficient HPO and NAS with progressive resource allocation. In *International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=syfgJE6nFRW.

Daniel Bolya, Rohit Mittapalli, and Judy Hoffman. Scalable diverse model selection for accessible transfer learning. In *Advances in Neural Information Processing Systems*, volume 34, pp. 19301–19312, 2021. URL https://proceedings.neurips.cc/paper/2021/hash/a1140a3d0df1c81e24ae954d935e8926-Abstract.html.

Xavier Bouthillier, Pierre Delaunay, Mirko Bronzi, Assya Trofimov, Brennan Nichyporuk, Justin Szeto, Nazanin Mohammadi Sepahvand, Edward Raff, Kanika Madan, Vikram Voleti, Samira Ebrahimi Kahou, Vincent Michalski, Tal Arbel, Chris Pal, Gael Varoquaux, and Pascal Vincent. Accounting for variance in machine learning benchmarks. In *Proceedings of Machine Learning and Systems*, volume 3, 2021. URL https://proceedings.mlsys.org/paper_files/paper/2021/hash/0184b0cd3cfb185989f858a1d9f5c1eb-Abstract.html.

Gavin C. Cawley and Nicola L. C. Talbot. On over-fitting in model selection and subsequent selection bias in performance evaluation. *Journal of Machine Learning Research*, 11:2079–2107, 2010. URL https://jmlr.org/papers/v11/cawley10a.html.

Janez Demšar. Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research*, 7(1):1–30, 2006. URL https://jmlr.org/papers/v7/demsar06a.html.

Tobias Domhan, Jost Tobias Springenberg, and Frank Hutter. Speeding up automatic hyperparameter optimization of deep neural networks by extrapolation of learning curves. In *Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence*, pp. 3460–3468, 2015. URL https://www.ijcai.org/Proceedings/15/Papers/487.pdf.

Romain Egele, Felix Mohr, Tom Viering, and Prasanna Balaprakash. The unreasonable effectiveness of early discarding after one epoch in neural network hyperparameter optimization. *Neurocomputing*, 597:127964, 2024. URL https://doi.org/10.1016/j.neucom.2024.127964.

Katharina Eggensperger, Philipp Müller, Neeratyoy Mallik, Matthias Feurer, René Sass, Aaron Klein, Noor Awad, Marius Lindauer, and Frank Hutter. HPOBench: A collection of reproducible multi-fidelity benchmark problems for HPO. In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks*, volume 1, 2021. URL https://datasets-benchmarks-proceedings.neurips.cc/paper_files/paper/2021/hash/93db85ed909c13838ff95ccfa94cebd9-Abstract-round2.html.

Stefan Falkner, Aaron Klein, and Frank Hutter. BOHB: Robust and efficient hyperparameter optimization at scale. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 1437–1446, 2018. URL https://proceedings.mlr.press/v80/falkner18a.html.

Matthias Feurer, Katharina Eggensperger, Stefan Falkner, Marius Lindauer, and Frank Hutter. Auto-Sklearn 2.0: Hands-free AutoML via meta-learning. *Journal of Machine Learning Research*, 23 (261):1–61, 2022. URL https://jmlr.org/papers/v23/21-0992.html.

C. A. Field and A. H. Welsh. Bootstrapping clustered data. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 69(3):369–390, 2007. URL https://doi.org/10.1111/j.1467-9868.2007.00593.x.

Zhongkai Hao, Chang Su, Songming Liu, Julius Berner, Chengyang Ying, Hang Su, Anima Anandkumar, Jian Song, and Jun Zhu. DPOT: Auto-regressive denoising operator transformer for large-scale PDE pre-training. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 17616–17635. PMLR, 2024. URL https://proceedings.mlr.press/v235/hao24d.html.

Maximilian Herde, Bogdan Raonić, Tobias Rohner, Roger Käppeli, Roberto Molinaro, Emmanuel de Bézenac, and Siddhartha Mishra. Poseidon: Efficient foundation models for PDEs. In *Advances in Neural Information Processing Systems*, 2024. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/84e1b1ec17bb11c57234e96433022a9a-Abstract-Conference.html.

Peiyan Hu, Haodong Feng, Hongyuan Liu, Tongtong Yan, Wenhao Deng, Tianrun Gao, Rong Zheng, Haoren Zheng, Chenglei Yu, Chuanrui Wang, Kaiwen Li, Zhi-Ming Ma, Dezhi Zhou, Xingcai Lu, Dixia Fan, and Tailin Wu. RealPDEBench: A benchmark for complex physical systems with real-world data. In *International Conference on Learning Representations*, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/aed13d8a5ee83a9c6880285b69d0ba81-Abstract-Conference.html.

Kosuke Imai and Michael Lingzhi Li. Experimental evaluation of individualized treatment rules. *Journal of the American Statistical Association*, 118(541):242–256, 2023. URL https://doi.org/10.1080/01621459.2021.1923511.

Kevin Jamieson and Ameet Talwalkar. Non-stochastic best arm identification and hyperparameter optimization. In *Proceedings of the 19th International Conference on Artificial Intelligence and Statistics*, volume 51 of *Proceedings of Machine Learning Research*, pp. 240–248, 2016. URL https://proceedings.mlr.press/v51/jamieson16.html.

Nathan Kallus. What's the harm? sharp bounds on the fraction negatively affected by treatment. In *Advances in Neural Information Processing Systems*, volume 35, 2022. doi: 10.52202/068431-1164. URL https://proceedings.neurips.cc/paper_files/paper/2022/hash/666ccccc6376058e251315b4de7e085b9-Abstract-Conference.html.

Pascal Kerschke, Holger H. Hoos, Frank Neumann, and Heike Trautmann. Automated algorithm selection: Survey and perspectives. *Evolutionary Computation*, 27(1):3–45, 2019. doi: 10.1162/evco_a_00242. URL https://arxiv.org/abs/1811.11597.

Felix Koehler, Simon Niedermayr, Rüdiger Westermann, and Nils Thuerey. APEBench: A benchmark for autoregressive neural emulators of PDEs. In *Advances in Neural Information Processing Systems*, volume 37, 2024. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/d9875ebcf74bccdc5076acab0dbee62c-Abstract-Datasets_and_Benchmarks_Track.html.

Liam Li, Kevin Jamieson, Afshin Rostamizadeh, Ekaterina Gonina, Jonathan Ben-tzur, Moritz Hardt, Benjamin Recht, and Ameet Talwalkar. A system for massively parallel hyperparameter tuning. In *Proceedings of Machine Learning and Systems*, volume 2, pp. 230–246, 2020. URL https://proceedings.mlsys.org/paper_files/paper/2020/hash/a06f20b349c6cf09a6b171c71b88bbfc-Abstract.html.

Lisha Li, Kevin Jamieson, Giulia DeSalvo, Afshin Rostamizadeh, and Ameet Talwalkar. Hyperband: A novel bandit-based approach to hyperparameter optimization. *Journal of Machine Learning Research*, 18(185):1–52, 2018. URL https://jmlr.org/papers/v18/16-558.html.

Zongyi Li, Nikola Kovachki, Kamyar Azizzadenesheli, Burigede Liu, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. Fourier neural operator for parametric partial differential equations. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=c8P9NQVtmnO.

Phillip Lippe, Bas Veeling, Paris Perdikaris, Richard Turner, and Johannes Brandstetter. PDE-Refiner: Achieving accurate long rollouts with neural PDE solvers. In *Advances in Neural Information Processing Systems*, volume 36, 2023. URL https://papers.neurips.cc/paper_files/paper/2023/hash/d529b943af3dba734f8a7d49efcb6d09-Abstract-Conference.html.

Anastasia Makarova, Huibin Shen, Valerio Perrone, Aaron Klein, Jean Baptiste Faddoul, Andreas Krause, Matthias Seeger, and Cedric Archambeau. Automatic termination for hyperparameter optimization. In *Proceedings of the First International Conference on Automated Machine Learning*, volume 188 of *Proceedings of Machine Learning Research*, pp. 7/1–21. PMLR, 2022. URL https://proceedings.mlr.press/v188/makarova22a.html.

Neeratyoy Mallik, Edward Bergman, Carl Hvarfner, Danny Stoll, Maciej Janowski, Marius Lindauer, Luigi Nardi, and Frank Hutter. PriorBand: Practical hyperparameter optimization in the age of deep learning. In *Advances in Neural Information Processing Systems*, volume 36, pp. 60066–60098, 2023. URL https://papers.neurips.cc/paper_files/paper/2023/hash/1704fe7aaff33a54802b83a016050ab8-Abstract-Conference.html.

Michael McCabe, Bruno Régaldo-Saint Blancard, Liam H. Parker, Ruben Ohana, Miles Cranmer, Alberto Bietti, Michael Eickenberg, Siavash Golkar, Geraud Krawezik, François Lanusse, Mariel Pettee, Tiberiu Tesileanu, Kyunghyun Cho, and Shirley Ho. Multiple physics pretraining for spatiotemporal surrogate models. In *Advances in Neural Information Processing Systems*, 2024. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/d7cb9db5ade2db7814fbd01ee59f4c7b-Abstract-Conference.html.

Nick McGreivy and Ammar Hakim. Weak baselines and reporting biases lead to overoptimism in machine learning for fluid-related partial differential equations. *Nature Machine Intelligence*, 6 (10):1256–1269, 2024. URL https://doi.org/10.1038/s42256-024-00897-5.

Felix Mohr and Jan N. van Rijn. Fast and informative model selection using learning curve cross-validation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(8):9669–9680, 2023. URL https://doi.org/10.1109/TPAMI.2023.3251957.

Felix Mohr and Jan N. van Rijn. Learning curves for decision making in supervised machine learning: A survey. *Machine Learning*, 113(11-12):8371–8425, 2024. URL https://doi.org/10.1007/s10994-024-06619-7.

Felix Mohr, Tom J. Viering, Marco Loog, and Jan N. van Rijn. LCDB 1.0: An extensive learning curves database for classification tasks. In *Machine Learning and Knowledge Discovery in Databases: ECML PKDD 2022*, volume 13717 of *Lecture Notes in Computer Science*, pp. 3–19. Springer, 2023. URL https://doi.org/10.1007/978-3-031-26419-1_1.

Michael Munn and Susan Wei. A bayesian model selection criterion for selecting pretraining checkpoints. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 45256–45271, 2025. URL https://proceedings.mlr.press/v267/munn25a.html.

Daniel Musekamp, Marimuthu Kalimuthu, David Holzmüller, Makoto Takamoto, and Mathias Niepert. Active learning for neural PDE solvers. In *International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/60960ad78868fce5c165295fbd895060-Abstract-Conference.html.

Thomas Nagler, Lennart Schneider, Bernd Bischl, and Matthias Feurer. Reshuffling resampling splits can improve generalization of hyperparameter optimization. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang (eds.), *Advances in Neural Information Processing Systems*, volume 37, pp. 40486–40533. Curran Associates, Inc., 2024. doi: 10.52202/079017-1281. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/47811ee68103bfcde7ca2223fccefb3a-Abstract-Conference.html.

Cuong Nguyen, Tal Hassner, Matthias Seeger, and Cedric Archambeau. LEEP: A new measure to evaluate transferability of learned representations. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 7294–7305, 2020. URL https://proceedings.mlr.press/v119/nguyen20b.html.

Cuong N. Nguyen, Phong Tran, Lam Si Tung Ho, Vu Dinh, Anh T. Tran, Tal Hassner, and Cuong V. Nguyen. Simple transferability estimation for regression tasks. In *Proceedings of the Thirty-Ninth Conference on Uncertainty in Artificial Intelligence*, volume 216 of *Proceedings of Machine Learning Research*, pp. 1510–1521. PMLR, 2023. URL https://proceedings.mlr.press/v216/nguyen23a.html.

Ruben Ohana, Michael McCabe, Lucas Meyer, Rudy Morel, Fruzsina J. Agocs, Miguel Beneitez, Marsha Berger, Blakesley Burkhart, Stuart B. Dalziel, Drummond B. Fielding, Daniel Fortunato, Jared A. Goldberg, Keiya Hirashima, Yan-Fei Jiang, Rich R. Kerswell, Suryanarayana Maddu, Jonah Miller, Payel Mukhopadhyay, Stefan S. Nixon, Jeff Shen, Romain Watteaux, Bruno Régaldo-Saint Blancard, François Rozet, Liam H. Parker, Miles Cranmer, and Shirley Ho. The Well: a large-scale collection of diverse physics simulations for machine learning. In *Advances in Neural Information Processing Systems*, volume 37, 2024. URL https://proceedings.nips.cc/paper_files/paper/2024/hash/4f9a5acd91ac76569f2fe291b1f4772b-Abstract-Datasets_and_Benchmarks_Track.html.

Isaac Ong, Amjad Almahairi, Vincent Wu, Wei-Lin Chiang, Tianhao Wu, Joseph E. Gonzalez, M. Kadous, and Ion Stoica. RouteLLM: Learning to route LLMs from preference data. In Y. Yue, A. Garg, N. Peng, F. Sha, and R. Yu (eds.), *International Conference on Learning Representations*, volume 2025, pp. 34433–34448, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/5503a7c69d48a2f86fc00b3dc09de686-Abstract-Conference.html.

Gabriel Peyré and Marco Cuturi. Computational optimal transport: With applications to data science. *Foundations and Trends in Machine Learning*, 11(5–6):355–607, 2019. doi: 10.1561/2200000073. URL https://www.nowpublishers.com/article/Details/MAL-073.

Florian Pfisterer, Lennart Schneider, Julia Moosbauer, Martin Binder, and Bernd Bischl. YAHPO Gym: An efficient multi-objective multi-fidelity benchmark for hyperparameter optimization. In *Proceedings of the First International Conference on Automated Machine Learning*, volume 188 of *Proceedings of Machine Learning Research*, pp. 3/1–39, 2022. URL https://proceedings.mlr.press/v188/pfisterer22a.html.

Sebastian Pineda Arango, Fabio Ferreira, Arlind Kadra, Frank Hutter, and Josif Grabocka. Quicktune: Quickly learning which pretrained model to finetune and how. In *International Conference on Learning Representations*, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/3812c3c803e986eca64000fe92467991-Abstract-Conference.html.

Herilalaina Rakotoarison, Steven Adriaensen, Neeratyoy Mallik, Samir Garibov, Eddie Bergman, and Frank Hutter. In-context freeze-thaw Bayesian optimization for hyperparameter optimization. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 41982–42008. PMLR, 2024. URL https://proceedings.mlr.press/v235/rakotoarison24a.html.

Rishabh Ranjan, Saurabh Garg, Mrigank Raman, Carlos Guestrin, and Zachary Lipton. Post-hoc reversal: Are we selecting models prematurely? In *Advances in Neural Information Processing Systems*, 2024. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/a6805b5564bd8d813a81c4b5a97e5ca6-Abstract-Conference.html.

Mahindra Singh Rautela, Alexander Most, Siddharth Mansingh, Bradley C. Love, Alexander Scheinker, Diane Oyen, Nathan Debardeleben, Earl Lawrence, and Ayan Biswas. MORPH: PDE foundation models with arbitrary data modality. arXiv preprint arXiv:2509.21670, 2025. URL https://arxiv.org/abs/2509.21670v4. Version 4, January 2026.

Cedric Renggli, André Susano Pinto, Luka Rimanic, Joan Puigcerver, Carlos Riquelme, Ce Zhang, and Mario Lučić. Which model to transfer? finding the needle in the growing haystack. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 9205–9214, 2022. doi: 10.1109/CVPR52688.2022.00899. URL https://openaccess.thecvf.com/content/CVPR2022/html/Renggli_Which_Model_To_Transfer_Finding_the_Needle_in_the_Growing_CVPR_2022_paper.html.

David Salinas, Matthias Seeger, Aaron Klein, Valerio Perrone, Martin Wistuba, and Cedric Archambeau. Syne tune: A library for large scale hyperparameter tuning and reproducible research. In *Proceedings of the First International Conference on Automated Machine Learning*, volume 188 of *Proceedings of Machine Learning Research*, pp. 16/1–23, 2022. URL https://proceedings.mlr.press/v188/salinas22a.html.

Lennart Schneider, Bernd Bischl, and Matthias Feurer. Overtuning in hyperparameter optimization. In *Proceedings of the Fourth International Conference on Automated Machine Learning*, volume 293 of *Proceedings of Machine Learning Research*, 2025. URL https://proceedings.mlr.press/v293/schneider25a.html.

Prabhant Singh, Sibylle Hess, and Joaquin Vanschoren. How NOT to benchmark your SITE metric: Beyond static leaderboards and towards realistic evaluation. In *International Conference on Learning Representations*, pp. 148906–148927, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/f0ebc318e2df08360b2df559e81602e5-Abstract-Conference.html.

Shashank Subramanian, Peter Harrington, Kurt Keutzer, Wahid Bhimji, Dmitriy Morozov, Michael W. Mahoney, and Amir Gholami. Towards foundation models for scientific machine learning: Characterizing scaling and transfer behavior. In *Advances in Neural Information Processing Systems*, volume 36, pp. 71242–71262, 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/e15790966a4a9d85d688635c88ee6d8a-Abstract-Conference.html.

Keita Sunada and Kohei Izumi. Optimal treatment assignment rules under capacity constraints, 2025. URL https://arxiv.org/abs/2506.12225. Preprint.

Kevin Swersky, Jasper Snoek, and Ryan P. Adams. Freeze-thaw Bayesian optimization. arXiv preprint arXiv:1406.3896, 2014. URL https://arxiv.org/abs/1406.3896.

Makoto Takamoto, Timothy Praditia, Raphael Leiteritz, Daniel MacKinlay, Francesco Alesiani, Dirk Pflüger, and Mathias Niepert. PDEBench: An extensive benchmark for scientific machine learning. In *Advances in Neural Information Processing Systems*, volume 35, 2022. URL https://proceedings.nips.cc/paper_files/paper/2022/hash/0a9747136d411fb83f0cf81820d44afb-Abstract-Datasets_and_Benchmarks.html.

Ioannis Tsamardinos, Elissavet Greasidou, and Giorgos Borboudakis. Bootstrapping the out-of-sample predictions for efficient and accurate cross-validation. *Machine Learning*, 107(12):1895–1922, 2018. doi: 10.1007/s10994-018-5714-4. URL https://link.springer.com/article/10.1007/s10994-018-5714-4.

Steve Yadlowsky, Scott Fleming, Nigam Shah, Emma Brunskill, and Stefan Wager. Evaluating treatment prioritization rules via rank-weighted average treatment effects. *Journal of the American Statistical Association*, 120(549):38–51, 2025. URL https://doi.org/10.1080/01621459.2024.2393466.

Kaichao You, Yong Liu, Jianmin Wang, and Mingsheng Long. LogME: Practical assessment of pre-trained models for transfer learning. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 12133–12143, 2021. URL https://proceedings.mlr.press/v139/you21b.html.

Guanhua Zhang, Ricardo Dominguez-Olmedo, and Moritz Hardt. Train-before-test harmonizes language model rankings. In *International Conference on Learning Representations*, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/79a97ba70ee6aff126f3090b10ad6702-Abstract-Conference.html.

Lucas Zimmer, Marius Lindauer, and Frank Hutter. Auto-PyTorch: Multi-fidelity metalearning for efficient and robust AutoDL. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 43 (9):3079–3090, 2021. URL https://doi.org/10.1109/TPAMI.2021.3067763.

AI USE STATEMENT

Generative AI tools were used for manuscript polishing, for literature search and retrieval of related work, and for writing and reviewing code. All retrieved references were checked against their original sources, and all content has been verified by the authors, who take responsibility for it.

A AUDIT INTERFACE AND REPRODUCTION

The CPU program evaluates the same report quantities from either of two inputs: complete candidate trajectories from which it reconstructs each route's return, or tables of already reconstructed returns. This appendix specifies both interfaces. Appendix H describes the training and data generation that produced the records.

A.1 INPUT TABLES

A trajectory row contains `context_id`, `candidate_id`, `step`, `validation_score`, `test_loss`, `cumulative_cost`, and `state_id`. The stopping-plan table lists the candidate's last executed step, final-retention status, and menu type for each route and context. Policy rows give route probabilities; context rows give weights, frequency groups, and outer cluster identifiers. Metadata declares the return rule, first legal return time, units, and the meaning of state equality.

A route can reuse a candidate's recorded trajectory when pruning leaves that candidate's data order, optimizer state, and learning-rate schedule unchanged. If the training procedure changes, the route needs records from that procedure. Cost is charged through the recorded stopping step, including training after a checkpoint that final validation later selects. Validation chooses the returned state before its test loss is read; ties are resolved by candidate identifier and then earlier update. Survivor-best searches eligible states of the final survivors, cap-only searches their stopping states, and all-reached includes states computed before a candidate was discarded. The program requires a cumulative cost at every executed stop and rejects duplicate identities, decreasing cumulative costs, missing selected outcomes, and incomplete menus.

A.2 REPORTS AND WEIGHTED AGGREGATION

Let group g have weight $W_g = \sum_{c \in g} w_c$. Normalize within that group as $w_{c|g} = w_c / W_g$ and compute $q_{gk} = \sum_{c \in g} w_{c|g} \pi_{ck}$. The report averages group gains and bounds as $\sum_g W_g M_g$ and $\sum_g W_g U_g$. Because frequencies must be preserved within every group, the optimization separates into one transport problem per group, and combining their optima attains the pooled bound. Each PDE is a group; the 25 LCBench audit datasets form one group.

The number of distinct returned states is counted over every route in the declared companion menu, including routes the evaluated policy never selects. The matching bound uses only routes with positive selection frequency. The returned-candidate cost share is weighted mean cost for that candidate divided by weighted mean total route cost, including any work after an earlier checkpoint is selected. Outputs contain the reconstructed states, route frequencies, per-context and per-group results, direct comparisons, and the finite-matrix assignment used to check $U(q)$.

A.3 SOURCE MAPPINGS AND IDENTITY CHECKS

The six-task PDE records contain 2,880 candidate runs for 480 contexts; external PDE records contain 720 runs for 120 contexts. Each normalized cross-initialization state uses four runs for mean-log validation and cost, and the held-out run for the tested state. The denominator is the study's full-set cross-initialization reference. Exact identity comparisons use model, update, and parameter hash. The companion routes screen all six candidates through update 100 and continue the selected pair to 3000.

The LCBench mapping uses the fixed donor ranking of 32 configurations on 25 audit datasets. Each configuration contributes one recorded run, with state identity given by dataset, configuration, and epoch. We convert validation and test accuracies to errors and retain cumulative recorded seconds as cost. The 31 companion routes share the first five epochs, so changing the return rule changes eligible states while keeping the early companion choices and executed training fixed.

The six-task and external adaptive reconstructions agree with all 600 recorded PDE actions, state hashes, losses, and training costs. LCBench reconstruction matches the fixed-design actions and enumerated companions. The code tests missing data, validation ties, cached return, complete continuation costs, signed gains, and finite transport on additive and nonadditive matrices. These checks validate the report implementation on the supplied data; the statistical units and hypotheses are specified in Appendix C.

A.4 Reported Extended Results

The supplement stores complete policy, budget, candidate-family, and recipe tables under `reported_results/`. A table index retains their captions, units, uncertainty, and comparison roles, with each table available as CSV and typeset source. The reproduction coverage distinguishes these stored results from estimates regenerated from the per-run records.

A.5 What the Code Reproduces

The CPU package reconstructs the three RPA reports, the fixed-design classification comparisons and return-rule diagnostics, compact recorded recipe comparisons, calibrated validation costs, and decomposition intervals. The coverage documentation distinguishes regenerated outputs from reported evidence. The full policy sweeps and their one-sided bounds, the 540-comparison family, and the donor-reconstruction sensitivity are documented by the tables or summary statements in this paper and the reported-result collection.

A.6 Execution

The supplement README gives the commands. A single CPU command runs the covered analyses and writes a PASS/FAIL table comparing each covered value with the number printed in this paper; the target file is used only for checking, never as an input to the estimates. The README also lists the pinned dependency versions and the coverage of each mode.

B Definitions, Matching, and Transport Diagnostics

B.1 Return Rules and Identity

For each context, the route menu supplies a returned loss and cost under the specified return rule. All routes use the same validation ordering and deterministic tie rule. A route may include validation-based pruning, in which case only states actually computed before its stopping decisions are available for return. The evaluated early policy uses its declared observations; the opportunity calculation instead uses the complete table of returned test losses.

Suppose the state selected by full-set validation remains in the retained candidates and is still eligible for return. It must also be the best state in that smaller set: a state ranked ahead of it there would also rank ahead in the full set. The two choices therefore return the same state when scores, eligibility, and tie order are unchanged.

For default-preserving pairs, suppose each candidate has the same admissible saved states regardless of its companion. With default d, let $\mathrm{rank}(d)$ be the rank of its best state among the N_{cand} candidate-best states under the common validation order. Each of the $\mathrm{rank}(d)-1$ better candidates returns its own state; every worse candidate returns the default's state. There are $(\mathrm{rank}(d)-1)+\mathbf{1}\{\mathrm{rank}(d)<N_{\mathrm{cand}}\}=\min(\mathrm{rank}(d),N_{\mathrm{cand}}-1)$ distinct pair outputs. This count explains how a default can merge companion choices while their training costs remain different.

Now add the same computed prefix to every route. If a_c^{pre} is the prefix's validation-best state, each survivor-best state a_{ck} becomes $\min(a_{ck},a_c^{\mathrm{pre}})$. Applying one map to the previous outputs cannot increase the number of distinct states. This explains the LCBench count comparison when the prefix is shared; different route-specific prefixes require their own state sets.

B.2 Usage, Matching, and PAPE

For one frequency group, let u_k be the uniform probabilities and $q_k=\sum_c w_c\pi_{ck}$ the adaptive route frequencies. Then

$$\bar{L}^u-\bar{L}^\pi=(\bar{L}^u-\bar{L}^q)+(\bar{L}^q-\bar{L}^\pi).$$

The first term is usage and the second is matching. To express the same split relative to any shared baseline L_{c0}, define $G_{ck} = L_{c0} - L_{ck}$. Expanding the expected gain gives

$$\bar{L}^u - \bar{L}^\pi = \sum_k (q_k - u_k)\mathbb{E}[G_k] + \sum_k \mathrm{Cov}(\pi_k, G_k).$$

The covariance sum measures matching: it is positive when the policy assigns more weight to a route on contexts where that route improves more over the baseline. The identity is exact for the specified finite weights, and any context-only baseline cancels from the comparison.

With two choices, write the potential outcome as $Y_c(k) = -L_{ck}$. Imai and Li's population average prescriptive effect compares the policy with independent assignment at the same treatment proportion (Imai & Li, 2023, Equation 2). Its finite-record form is therefore the binary matching term in Equation 1. RPA applies that term to several training routes, including routes that return the same state at different costs. Complete records allow each context's random-control expectation to be summed directly. Appendix C describes uncertainty over new contexts using the study's sampling groups.

When the early policy and its frequencies are fixed and return rule r changes to s, matching changes by

$$M_s - M_r = (\bar{L}^s_q - \bar{L}^r_q) - (\bar{L}^s_\pi - \bar{L}^r_\pi).$$

Allowing more saved states can therefore improve both workflows while reducing the mean difference between them. The paired observations and their intervals determine whether that change is supported in a particular comparison. Define the best returned loss as $L^*_r = \min_{\gamma \in \Pi(w,q)} \langle \gamma, L^r \rangle$. Since $U_r = \bar{L}^r_q - L^*_r$ and $M_r = \bar{L}^r_q - \bar{L}^r_\pi$,

$$U_r - M_r = \bar{L}^r_\pi - L^*_r.$$

This exact accounting relates the two audit quantities to the policy's excess loss above the rule-specific, test-informed optimum. The optimum and the loss matrix both change with r; Table 4 reports that relationship on the fixed classification design.

B.3 What Complete Route Rows Identify

For a fixed collection of records, return rule, and policy, the complete loss matrix L determines each context's random-control expectation and the optimum in Equation 2. Summation gives the former, and a finite linear program gives the latter. Both calculations hold the observed loss rows and fitted frequencies fixed. Uncertainty over new contexts or a newly constructed candidate pool is handled by the statistical analysis.

The important information is how alternative outcomes occur together for the same instance. In a treatment experiment with one observed outcome per unit, randomization can identify average policy values while leaving the dependence between that unit's potential outcomes unknown. Kallus studies the resulting uncertainty in the fraction harmed by treatment (Kallus, 2022, Sections 2–3). The example below shows why the same missing dependence matters for the best assignment with full outcome information.

Let the two row types in each matrix have equal weight and be unobserved in the available covariates. Their potential losses are

$$L^{\text{aligned}} = \begin{pmatrix} 0 & 0 \\ 1 & 1 \end{pmatrix}, \qquad L^{\text{crossed}} = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, \qquad w = q = (1/2, 1/2).$$

Observing one randomly assigned route per unit gives the same outcome distribution in both cases: each route has loss zero or one with equal probability. The independent-control loss is $1/2$ for both. In the aligned matrix, every feasible assignment has mean loss $1/2$, so the same-frequency minimum is $1/2$ and $U(q) = 0$. In the crossed matrix, assigning each row to its zero-loss route preserves q, attaining minimum loss zero and $U(q) = 1/2$. Complete rows distinguish these cases; route marginals alone do not.

The optimal assignment knows each unit's full loss row. A policy seeing only constant covariates faces equal conditional mean losses for the two routes and has zero matching opportunity, as formalized below. Paired model-response records and algorithm-selection benchmarks can also expose alternative outcomes (Ong et al., 2025; Kerschke et al., 2019); here the loss row is obtained by reconstructing which state validation returns under each training plan.

B.4 Matching Bound and Equality Conditions

We prove Proposition 1 on the finite allocation set $\Pi(w, q)$ of Equation 2. Remove zero-frequency routes and assume positive context weights, finite losses, and a feasible complete rectangle. The product allocation is feasible, so $U(q) \geq 0$. Any policy at those frequencies has joint weights $w_c\pi_{ck} \in \Pi(w, q)$ and hence matching gain at most $U(q)$. A minimizer exists by compactness; selecting route k with conditional probability γ^*_{ck}/w_c attains it with full matrix knowledge.

If $L_{ck} = \alpha_c + \beta_k$, every feasible allocation has loss $\sum_c w_c\alpha_c + \sum_k q_k\beta_k$, so $U = 0$. Conversely, if $U = 0$, the strictly positive product allocation minimizes a linear objective. In any two-row, two-column rectangle, both signs of a sufficiently small mass exchange preserve feasibility. Optimality forces

$$L_{ck} + L_{dl} - L_{cl} - L_{dk} = 0.$$

Fixing c_0, k_0 and setting $\alpha_c = L_{ck_0}$ and $\beta_k = L_{c_0k} - L_{c_0k_0}$ produces the additive representation. The exchange condition gives the claimed zero-opportunity equivalence for the returned-loss matrix.

Subtracting a term that depends only on context, or only on route, leaves M and U unchanged because their marginal weights are fixed. Even a route that performs better everywhere can support matching when the size of its advantage varies across contexts. Conversely, many distinct returned states can have an additive loss matrix and offer no gain from reassignment. If some context–route cells are forbidden, their weights are fixed to zero and only exchanges along feasible cycles remain available.

B.5 Information Available to Selection

Let a finite early signal X have positive probabilities μ_x and define $\eta_{xk} = \mathbb{E}[L_k \mid X = x]$. Candidate outcomes may be dependent. A policy uses X and independent randomization, so its loss is $\sum_x \mu_x \sum_k \pi_k(x)\eta_{xk}$. Its joint allocation ranges over exactly $\Pi(\mu, q)$ when marginal usage is fixed. Equation 2 applied to this conditional matrix gives the information-constrained opportunity, and zero opportunity corresponds to an additive η.

This conditional loss table gives the matching opportunity accessible to the stated early signal. The complete test-loss table instead gives the opportunity available with all recorded outcomes in hand. Estimating conditional losses on held-out instances would assess how much of the latter opportunity the early signal reveals.

B.6 Heterogeneous Route Costs

For finite costs C_{ck} and a feasible average allowance $\mathcal{B}$, define

$$V_{\mathcal{B}} = \min_{\gamma \in \Pi(w,q),\, \langle \gamma, C\rangle \leq \mathcal{B}} \langle \gamma, L\rangle = \max_{\lambda \geq 0} \left\{ \min_{\gamma \in \Pi(w,q)} \langle \gamma, L + \lambda C\rangle - \lambda\mathcal{B} \right\}. \tag{4}$$

Attach a multiplier $\lambda \geq 0$ to the budget and potentials f_c, g_k to the margins. The Lagrangian infimum is finite when $f_c + g_k \leq L_{ck} + \lambda C_{ck}$. The dual objective is $\sum_c w_c f_c + \sum_k q_k g_k - \lambda\mathcal{B}$. Strong finite linear-programming duality applies to the nonempty compact primal (Peyré & Cuturi, 2019); for fixed λ, transport duality gives the inner minimum in Equation 4.

The gain bound at budget $\mathcal{B}$ is $\langle wq^\top, L\rangle - V_{\mathcal{B}}$. It is nonnegative when the independent control is within budget; a tighter budget may require a higher loss. Replacing losses and costs by their conditional means given X gives the information-constrained version. A donor-trained static mixture is evaluated at its actual audit cost. Capacity-constrained treatment assignment also uses optimal transport (Sunada & Izumi, 2025), but learns an assignment rule, whereas this finite calculation measures opportunity in the recorded outcomes.

C Comparison Definitions and Statistical Analysis

C.1 Selection Scores and References

The *four-system study* compares MPP-Tiny, Poseidon-T, Poseidon-B, and MORPH-Tiny on the same six PDEs as the six-task study: coupled Cahn–Hilliard, Brusselator, decaying Navier–Stokes, two-layer quasi-geostrophic dynamics, nonlinear Klein–Gordon, and nonlinear Schrödinger. It contributes to the adaptation evidence used to specify the fixed defaults described in Appendix D.2.

A PDE context comprises equation, dataset seed, supervision regime, and evaluated initialization. Cross-initialization validation averages log error from the other four runs; same-run validation uses the evaluated run. Both screening and final return follow the chosen rule. The four-system and six-task studies require positive finite validation errors; the external study additionally admits exact zero by replacing it with 10^{-30} inside the logarithm. Invalid scores receive $+\infty$. Candidate identifier and then earlier update break ties.

For each PDE context, the common reference is the state chosen from the full candidate pool by cross-initialization validation. Its test error $D_c > 0$ remains fixed in the paired recipe comparisons. Tables marked own-reference instead use the full-set state from the selection rule or recipe named in that table. With test error e_c, the common-reference gap is $e_c/D_c - 1$. Validation determines the reference state; test error supplies its denominator. Changing that denominator rescales the context's gap and can change pooled comparisons, so each main-text paired comparison keeps one reference throughout.

C.2 What an Early Curve Can Support

Learning-curve decisions may vary training updates or the number of training examples; these axes define different resource questions (Mohr & van Rijn, 2024). Our route reconstruction fixes the source's axis and stopping steps. Learning Curve Cross-Validation prunes candidates judged unlikely to remain competitive as sample sizes grow (Mohr & van Rijn, 2023); Makarova et al. stop optimization using the relation between remaining optimization error and statistical uncertainty (Makarova et al., 2022). These procedures motivate alternative choices of early information or stopping plans. Applying RPA requires the resulting routes' observed states and costs under the same declared return rule.

C.3 Independent Units and Fixed-Task Aggregation

The six-task and four-system studies each have eight dataset seeds per PDE, two supervision regimes, and five initialization views. A seed's ten contexts remain together. The external study has twelve seeds per PDE and five views; recipe studies use a matched eight-seed subset with the same five views. We first average over the dependent views within a seed and then weight PDE means equally. The number of recorded candidates, saved updates, enumerated routes, or bootstrap draws does not increase the independent task or seed count.

Repeated views of a seed share the same sampled data. Resampling whole clusters preserves this dependence, following the principle studied by Field & Welsh (2007). Classification comparisons similarly treat datasets as experimental units, rather than treating each model fit as a new task (Demšar, 2006). We estimate paired mean-loss differences, rather than a test of mean algorithm ranks.

For LCBench, each audit dataset is one outer unit. Resampling carries all policies and return rules for that dataset together. Random pool draws in an expanded setting are averaged within dataset before inference. Fixed-design resampling conditions on the donor-built pool and default; full donor/audit refitting repeats their construction and the fitted controls. This distinction follows the source of variation being evaluated: data, initialization, and hyperparameter construction can contribute different components to benchmark uncertainty (Bouthillier et al., 2021).

C.4 Analysis Roles

Table 5 states what each comparison tests and which analysis role it retains. Predeclared comparisons evaluate their specified designs; post-outcome extensions ask additional questions of the same records. Recipe comparisons remain paired on the same contexts.

C.5 Intervals for the Audit Decompositions

The decomposition intervals use 20,000 fixed-seed bootstrap draws within PDE, refitting each task's candidate frequencies in every draw. Table 6 gives both the pointwise 95% diagnostic intervals and 97.5% sensitivity endpoints from the same draws. The six-task negative matching estimate narrowly excludes zero at 95%, but not at 97.5%. These fixed-menu diagnostics have the role stated in Table 5; the LCBench primary intervals remain at 97.5%.

The external study's two predeclared adaptive comparisons use 97.5% marginal intervals. Other paired policy comparisons use their stated 95% intervals.

Table 5: Analysis roles and nominal interval coverage. Candidate frequencies are refitted in each resample when estimated from evaluation instances.

Comparison	Design role	Uncertainty and use
PDE usage/matching	Post-outcome fixed-menu diagnostic	Pointwise 95%; 97.5% sensitivity from the same 20,000 cluster draws.
External adaptive versus T and uniform	Predeclared paired comparisons	97.5% marginal intervals; other policy comparisons use 95%.
Unrestricted ranked pairs	Predeclared directional sequence	One-sided 95% bounds: Top-2 versus Top-1, then uniform pairs.
Calibrated-objective policies	Declared before treatment outcomes	Two 97.5% pooled policy contrasts; task rows describe their contributions.
Low-rate MPP-L models	Declared paired-model endpoints	Two 97.5% intervals for $R^+ - R$ and $N^+ - N$ after validation selects each state.
LCBench fixed design	Locked before audit-policy analysis	Two 97.5% intervals, conditional on donor pool/default.
Return rules and pool sensitivity	Post-exposure extensions	Paired 95% and four-contrast simultaneous intervals; 540-comparison family across expanded settings.
Donor/audit refit	Construction sensitivity	2,000 whole-pipeline draws; 97.5% percentile intervals.

Table 6: PDE decomposition sensitivity in relative-gap pp. Both coverage levels use the same 20,000 seed-cluster draws and refit frequencies within each draw.

Study	Gain	Estimate	95%	97.5%
Six-task PDE	Total	-0.004	$[-0.841, 0.781]$	$[-0.965, 0.886]$
	Usage	$+0.592$	$[-0.048, 1.145]$	$[-0.143, 1.231]$
	Matching	-0.596	$[-1.097, -0.021]$	$[-1.160, 0.036]$
External PDE	Total	$+1.621$	$[0.670, 2.578]$	$[0.537, 2.718]$
	Usage	$+1.409$	$[0.581, 2.313]$	$[0.468, 2.438]$
	Matching	$+0.212$	$[-0.168, 0.578]$	$[-0.228, 0.639]$

For the unrestricted shortlist comparisons, each study's serial directional test first compares ranked Top-2 with Top-1, then with uniform pairs; the reported one-sided 95% bounds retain that order. The role of each comparison is stated in Table 5; the extended-result archive retains its full values.

C.6 Validation Selection and Evaluation Target

Validation selects the returned state; held-out test loss evaluates that choice. Finite validation criteria can themselves be overfit by model selection (Cawley & Talbot, 2010; Schneider et al., 2025). Repeated reuse or reshuffling of splits changes that selection problem (Nagler et al., 2024). RPA retains the declared validation procedure and its test-evaluated result, rather than substituting a test-best state. BBC-CV estimates winner-score bias by resampling stored out-of-sample predictions (Tsamardinos et al., 2018); our cluster resampling estimates the paired usage and matching quantities for the recorded return procedure.

C.7 Small-Cluster Sensitivity and Random Execution

The external quality sensitivity averages paired differences within each seed and uses a Welch–Satterthwaite approximation to the equal-PDE mean. If task p has seed-level variance $\hat{\sigma}_p^2$ and n_p seeds, with P fixed tasks,

$$\widehat{\mathrm{SE}}^2 = P^{-2} \sum_p \hat{\sigma}_p^2 / n_p, \qquad \nu = \frac{(\sum_p \hat{\sigma}_p^2/n_p)^2}{\sum_p (\hat{\sigma}_p^2/n_p)^2/(n_p - 1)}.$$

Table 7 compares this approximate sensitivity with the prespecified percentile intervals.

Table 7: External cross-initialization quality sensitivity, adaptive companion minus comparator in relative-nRMSE-gap pp. Each comparison has 120 contexts in 24 seed clusters. The seed-level t intervals give approximate sensitivities at the same stated coverage.

Comparator	Difference	Coverage	Percentile	Approximate t
Fixed T	−2.250	97.5%	[−3.50, −1.05]	[−3.666, −0.835]
Uniform companion	−1.621	97.5%	[−2.71, −0.54]	[−2.889, −0.353]
Top-2 at 100	−7.774	95%	[−18.30, −0.34]	[−17.960, 2.413]

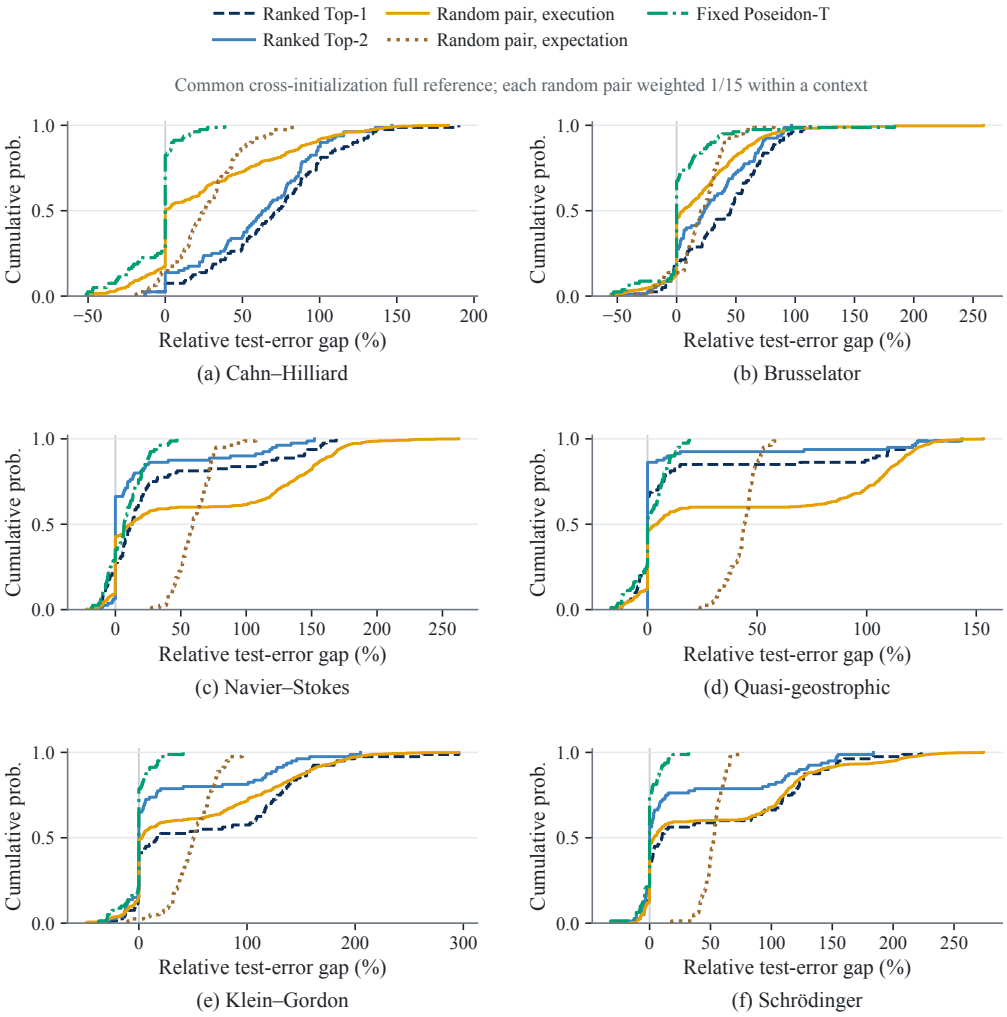


Figure 6: Relative test-error gaps in the six-task PDE study, 80 cross-initialization contexts per task. Execution keeps each context's 15 random-pair outcomes; expectation averages them first. Ranked Top-1 and Top-2 are the unrestricted shortlists.

Figure 6 compares two distributions for the six-task study. The execution distribution retains each context's 15 possible random-pair outcomes; the expected-decision distribution replaces them with their context mean. Averaging removes within-context selection variability and preserves the overall mean, but can change tail probabilities and quantiles. A direct random pair and a pair drawn after screening return the same predictions under survivor-best selection. Under all-reached selection, the additional full-pool prefix can change the return. Each retained tail table specifies which distribution and return rule it uses.

PDE main and cost analyses use the stated linear-interpolated quantile convention unless a table explicitly specifies the left-inverse weighted quantile for execution distributions. In the classification return-rule sensitivity, simultaneous intervals standardize each draw's centered deviation by that contrast's fixed bootstrap standard deviation and take the maximum over the four prespecified diagnostic contrasts. The center is the bootstrap mean. Full-pipeline and broad-family inference is described with the classification results in Appendix F.

D PDE Report and Comparator Evidence

Figure 7 reports absolute selected-policy errors for all six tasks, complementing the relative gaps. Figure 8 shows the external study's companion choices and context-level prediction changes. These views retain the heterogeneity behind the pooled report.

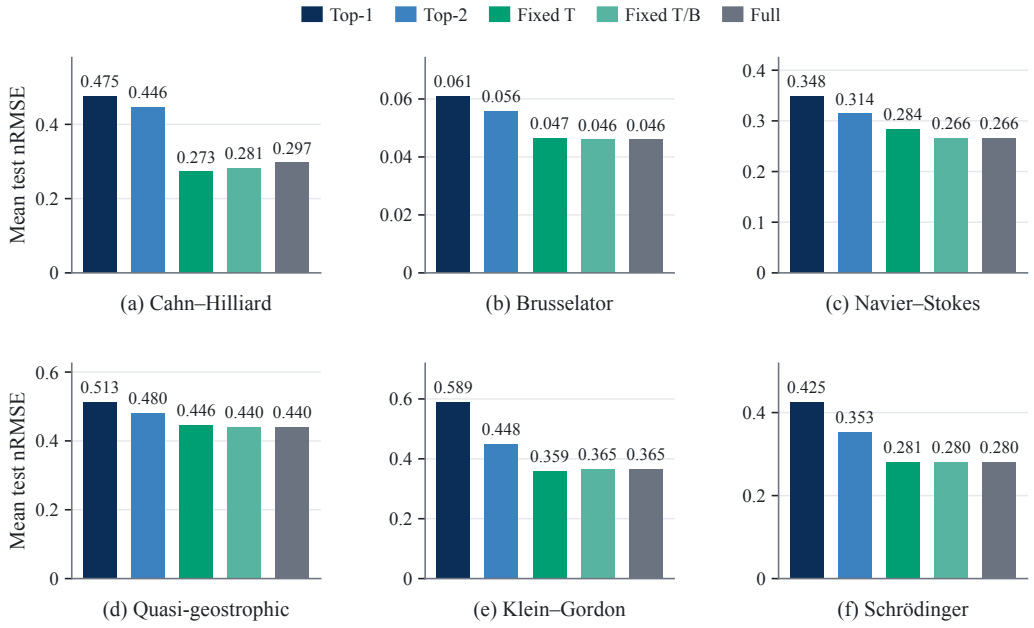


Figure 7: Absolute test nRMSE in the six-task PDE study under cross-initialization selection. Each panel compares unrestricted Top-1 and Top-2 at update 100, fixed T and T/B, and full-pool return. Task means use the same selection and scoring as the relative-gap analysis.

D.1 Complete Audit Groups and Named Direct Models

Tables 8 and 9 resolve the audit into its fixed task groups. The direct T/B means explain why an appropriate low-cost comparator differs between the six-task and external populations.

D.2 Defaults and Source Evidence

A fixed policy trains its declared identity set through $T_{\max}$ and applies the same return rule within that set. Poseidon-T, Poseidon-T/B, the largest model, and the largest pair were specified before the six-task PDE study's test outcomes. T and T/B were motivated by adaptation evidence from the same PDE families, then evaluated on new generated instances and initialization seeds. The costs of acquiring that evidence are distinct from adaptation to a new instance.

The additional default-source comparisons ask whether a preference learned on other PDE evidence transfers to a target task. Their role is diagnostic (Table 5); they do not enter the construction of the predeclared T/TB baselines above.

Table 8: Per-group RPA diagnostics. Gains and $U(q)$ use relative-nRMSE-gap pp for the PDE groups and classification-error pp for LCBench. Within each PDE, all regimes and initialization views contribute to the displayed fixed-record means. $U(q)$ uses test-informed reassignment at fixed within-group frequencies.

Group	Actions	Total	Usage	Matching	$U(q)$
Brusselator	1.48	−1.208	−0.955	−0.252	3.373
Cahn–Hilliard	1.46	−1.122	+0.642	−1.764	0.667
Navier–Stokes	1.95	+2.526	+2.807	−0.280	5.497
Klein–Gordon	1.44	−0.819	−0.134	−0.684	2.920
Schrödinger	1.48	+0.159	+0.128	+0.031	2.133
Quasi-geostrophic	1.74	+0.441	+1.065	−0.623	2.087
CGL	1.53	+0.211	+0.107	+0.103	1.454
MHD	2.00	+3.032	+2.711	+0.321	3.816
LCBench	8.68	+0.328	+0.078	+0.250	0.799

Table 9: Direct T and B, task-specific mean gap (%) to each study's cross-initialization full-set reference. The last column is B minus T in pp. All contexts are retained; this is a descriptive regrouping of the reported direct policies.

Task	Fixed T	Fixed B	Difference
Cahn–Hilliard	−4.96	4.05	+9.01
Brusselator	5.46	18.25	+12.80
Navier–Stokes	8.02	−0.75	−8.77
Quasi-geostrophic	1.70	−1.26	−2.96
Klein–Gordon	−0.72	−1.05	−0.34
Schrödinger	0.64	0.88	+0.24
MHD	6.02	0.00	−6.02
CGL	0.27	−0.28	−0.55

Leave-One-PDE-Out Validation. The validation-only leave-one-PDE-out rule omits the target PDE and scores each fixed subset on the other five PDEs. In each donor context, it subtracts the full pool's best admissible mean-log validation score from the subset's best score. The rule averages those differences within each donor PDE and then equally across donor PDEs, choosing the lowest-scoring subset with lexicographic ties. Target-PDE validation is used subsequently to select a saved state within the chosen set; it does not enter default identity selection.

All six held-out-PDE folds choose T for width one and T/B for width two. The reported intervals hold these selected defaults fixed and resample the target dataset-seed clusters. The six-task report uses the named defaults specified above, independently of this leave-one-PDE-out selection.

D.3 Unrestricted Rankings and Default-Preserving Choices

The unrestricted six-task Top-2 policy improves on Top-1 and uniform pairs under the stated directional tests. Default-preserving selection instead keeps T and asks which companion to assign. Table 10 gives the unrestricted contrasts, including the four-system study. Table 11 tests reduced candidate families using the reference and coverage specified for each reduced pool.

Table 10: Prespecified equal-PDE mean contrasts. Negative differences favor ranked Top-2; a negative one-sided upper bound supports the directional comparison. Each study has 480 contexts in 48 seed clusters.

Study	Ranked Top-2 minus	Difference (pp)	Upper 95% (pp)
Four systems	Top-1	−26.88	−19.71
	Random Top-2	−6.62	−3.51
Six systems	Top-1	−18.15	−13.97
	Random Top-2	−14.00	−7.73

Table 11: Recorded family-exclusion sensitivity, 480 contexts and 48 clusters per row. Differences are pp under each reduced pool's own reference; intervals are pointwise 95%. LOPO intervals condition on selected identities.

Removed	Top-2 − Top-1	Top-2 − random pair	LOPO single − Top-2
Poseidon	-3.58 $[-5.10, -2.28]$	-1.41 $[-2.93, 0.24]$	-3.13 $[-5.02, -1.28]$
MPP	-15.32 $[-24.87, -7.53]$	-5.07 $[-10.31, 0.68]$	-11.67 $[-18.34, -5.45]$
MORPH	-23.13 $[-30.16, -16.79]$	2.48 $[-3.13, 8.60]$	-14.32 $[-21.42, -7.79]$

D.4 External Policy Levels and Paired Comparisons

The two external equations each have 60 contexts in twelve seed clusters. Final-survivor policies all select saved states from update 100 onward. A drawn pair before screening omits other candidates' prefixes; after-screening drawing computes those prefixes. Under survivor-best return their selected actions agree but their work differs. Table 12 gives cross-initialization levels for the full competing menu. Table 13 gives the paired uncertainty, with the two predeclared comparisons retaining 97.5% intervals. Table 14 reports direct B, and Table 15 retains the staged comparison. The full-set reference uses B in 92 cross-initialization contexts and T in 28; same-run selection uses B in 72 and T in 48. Fixed T/B reproduces each rule's full-set state in all 120 contexts. Comparisons of different final identities retain their measured prediction differences, even when a direct route is cheaper.

Table 12: External cross-initialization levels, 120 contexts and 24 clusters. Common-gap intervals are pointwise 95%. L denotes MPP-L; L/B is the largest-parameter pair.

Policy	Common gap [95% CI] (%)	Own gap (%)	nRMSE
Fixed T	3.15 [2.26, 4.05]	3.15	0.4398
Fixed T/B	0.00 [0.00, 0.00]	0.00	0.4288
Largest single (L)	93.48 [89.31, 97.55]	93.48	0.8144
Largest pair (L/B)	–0.14 [–0.78, 0.49]	–0.14	0.4278
Top-1 at 100	30.47 [14.99, 47.43]	30.47	0.5539
Top-2 at 100	8.67 [0.98, 19.81]	8.67	0.4645
Random pair	37.51 [35.81, 39.16]	37.51	0.5833
T+validation	0.90 [0.22, 1.73]	0.90	0.4317
T+random before	2.52 [1.81, 3.24]	2.52	0.4376
T+random after	2.52 [1.81, 3.24]	2.52	0.4376
Top-1 at 600	1.46 [0.50, 2.43]	1.46	0.4340
Top-1 at 900	0.26 [–0.55, 1.12]	0.26	0.4293
Stage 100/600	5.98 [0.72, 15.42]	5.98	0.4515
Stage 300/900	0.26 [–0.55, 1.12]	0.26	0.4293
Full set	0.00 [0.00, 0.00]	0.00	0.4288

Table 13: External cross-initialization paired contrasts, 120 matched contexts in 24 seed clusters. The two primary rows also give 97.5% marginal intervals; other rows use 95% intervals. Negative values favor the first policy.

Comparison (first minus second)	Difference [95% CI] (pp)	97.5% CI (pp)
T+validation − Fixed T	–2.25 [–3.35, –1.19]	[–3.50, –1.05]
T+validation − T+random after	–1.62 [–2.58, –0.67]	[–2.71, –0.54]
T+validation − T+random before	–1.62 [–2.58, –0.67]	—
T+validation − Top-2 at 100	–7.77 [–18.30, –0.34]	—
T+validation − Fixed T/B	0.90 [0.22, 1.73]	—
Top-2 at 100 − Top-1 at 100	–21.80 [–36.06, –9.11]	—
Top-2 at 100 − Random pair	–28.84 [–36.51, –17.99]	—
Top-1 at 600 − Top-2 at 100	–7.21 [–18.63, 0.53]	—
Top-1 at 900 − Top-2 at 100	–8.41 [–19.58, –0.78]	—
Stage 100/600 − Top-2 at 100	–2.69 [–7.58, 1.12]	—
Stage 300/900 − Top-2 at 100	–8.41 [–19.58, –0.78]	—

Table 14: Direct-B replay on the external PDE study. Gap is relative to the common full-set reference; Δ denotes T+validation minus direct B. Task rows have 60 contexts and twelve seed clusters; all-task rows have 120 and 24. Entries are descriptive means and paired point differences. Work is GPU-s under the row's selection rule.

Selection	Task	B gap (%)	B work	Δ gap (pp)	Δ work
Cross-initialization	MHD	0.00	7,602	1.79	1,870
	CGL	−0.28	7,596	0.28	2,394
	All tasks	−0.14	7,599	1.04	2,132
Same-run	MHD	−0.61	1,900	1.66	397
	CGL	−0.11	1,899	−1.88	511
	All tasks	−0.36	1,900	−0.11	454

Table 15: Direct staged comparison, using 20,000 paired seed-cluster draws. Negative differences favor stage 300/900. Error and work are separate outcomes.

Information	Metric	Stage 300/900 minus T+validation [95% CI]
cross-initialization	Gap (pp)	−0.64 [−1.81, 0.50]
cross-initialization	Work (GPU-s)	−655 [−1,712, 463]
same-run	Gap (pp)	−0.71 [−1.90, 0.40]
same-run	Work (GPU-s)	−144 [−317, 29]

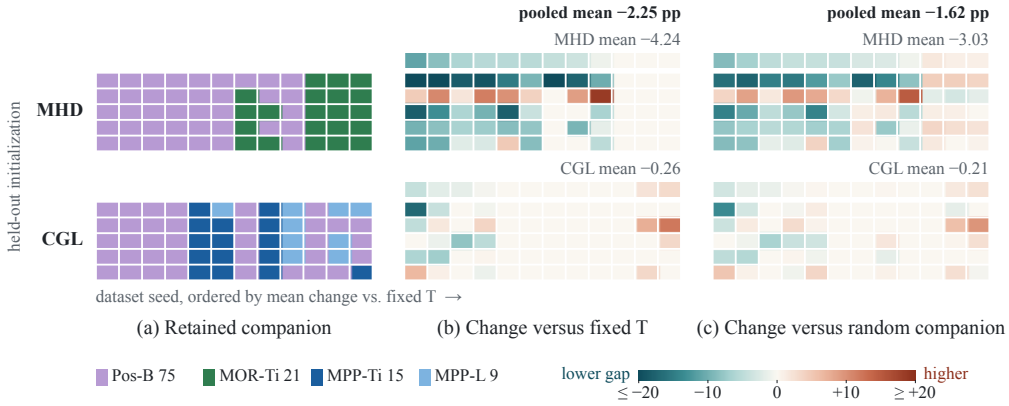


Figure 8: External PDE contexts under cross-initialization selection. Panels show the retained companion and returned-gap differences versus fixed T and versus a uniform companion. Columns are dataset seeds ordered by mean change versus T; rows are initialization views. Each task has twelve seed clusters and five views.

The archived tables give same-run levels, paired differences, task results, and random-execution tails for the same 120 contexts and common denominator.

E SENSITIVITY TO FITTED PDE RECIPES

The recipe comparisons reuse eight external dataset seeds per task, five initialization views, and 80 matched contexts. They change MPP recipes while reusing Poseidon and MORPH trajectories. Baseline training uses physical-space raw MSE. The raw and normalized calibrated variants tune MPP-Ti and MPP-L on independent development pairs. The lower-rate extensions change MPP-L only; they extend the same rate grid and reuse the identical raw-MHD runs when the selected rate is unchanged. Table 16 gives all five fitted settings. The common R/N grid is $\{2.5, 5, 10, 20, 40\} \times 10^{-4}$; its extension adds $\{0.25, 0.75\} \times 10^{-4}$ for MPP-L. Table 17 gives the two primary lower-rate endpoints. The normalized loss divides squared error by target energy with the same numerical floor as evaluation. All settings retain the baseline cross-initialization full-set denominator for common-gap comparisons. Same-run selection uses each evaluated run's validation; cross-initialization selection uses the other four runs for both screening and final return. Recipe settings are paired alternatives on the same dataset seeds, not additional independent populations.

Table 16: MPP trunk and I/O learning rates (equal within a run), in units of 10^{-4}. O is baseline; R/N use recalibrated raw/normalized objectives; R^+/N^+ extend the MPP-L grid. MPP-Ti is unchanged by the extensions.

System	Task	O	R	N	R^+	N^+
MPP-Ti	MHD	10	5	10	5	10
MPP-Ti	CGL	20	20	10	20	10
MPP-L	MHD	10	10	2.5	10	0.75
MPP-L	CGL	10	5	2.5	0.75	0.25

Table 17: Lower-rate extension minus the preceding calibrated recipe for MPP-L. Geometric test-nRMSE change after same-run validation selection; 80 paired contexts in 16 seed clusters. Protocol-defined 97.5% marginal intervals are retained.

Comparison	Change (%)	Interval (%)
$R^+ - R$	-0.953	$[-3.528, 1.742]$
$N^+ - N$	$+1.402$	$[-1.364, 4.915]$

E.1 MODEL RESPONSE AND POLICY CONSEQUENCE

Tables 18 and 19 retain the calibrated-objective policy endpoints and their task contributions. These compare policy outcomes; the lower-rate endpoints above compare the selected MPP-L model itself. Each uses its specified paired cluster resampling and coverage. A model change need not reach the full-set predictor or the default-preserving route.

Table 18: Prespecified same-run recipe contrasts on 80 matched contexts in 16 seed clusters. Differences are common-reference gap differences (pp); negative values favor the first policy.

Comparison	Difference	97.5% marginal interval
N T+validation minus R T+validation	$+0.212$	$[0.000, 0.637]$
N T+validation minus fixed B	$+0.306$	$[-1.000, 1.770]$

Across the five fitted settings, full-set returned identities, updates, and parameter hashes agree context by context under each selection rule. Every non-B default pair returns the fixed-T state. Figure 9 and Tables 20 and 21 locate changes in retention and final action. An early competitor can alter which other model survives an intermediate pruning step even when it never supplies the full-set reference.

Table 19: Task components of the two same-run primary comparisons. Each row has 40 contexts in eight clusters; differences are pp. Task-specific rows describe the two pooled primary contrasts.

Comparison	Task	Difference	97.5% marginal interval
N minus R, T+validation	MHD	0.000	$[0.000, 0.000]$
	CGL	0.424	$[0.000, 1.273]$
N T+validation minus fixed B	MHD	1.783	$[-0.340, 4.150]$
	CGL	-1.172	$[-2.598, 0.529]$

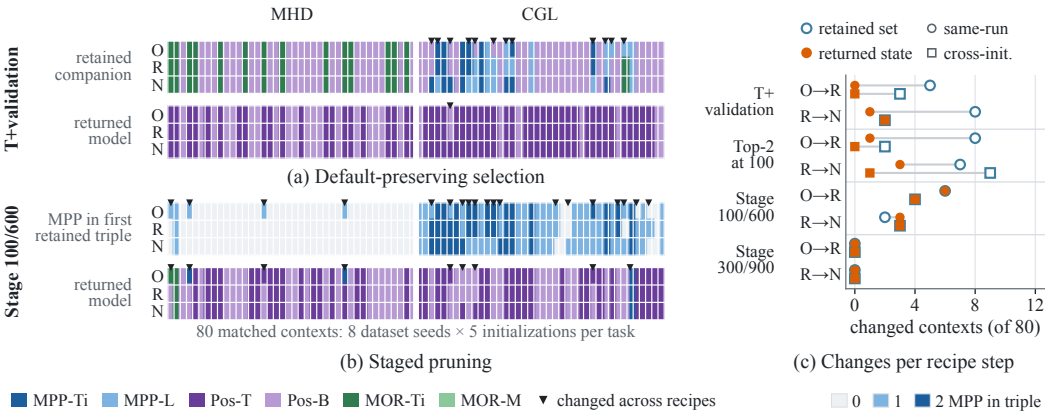


Figure 9: Recipe changes on 80 matched contexts. Rows O, R, and N are the baseline, recalibrated-raw, and recalibrated-normalized MPP recipes; triangles mark differences across recipes. Panels (a,b) use same-run selection: (a) T+validation companion and returned model; (b) Stage 100/600 MPP count in the first retained triple and returned model. (c) Changed retained sets (open) and returned model–update–states (filled), under both selection rules. Retention and return counts are in Table 20.

The T-fallback variant selects from companions with finite screening scores and continues T alone when that set is empty. Every context has an eligible companion in all five fitted settings, under both selection and final-state eligibility rules. The fallback is never triggered, so its returned states, losses, and costs equal ordinary T+validation in these records.

E.2 Direct Alternatives and Stopping Replay

Table 22 reports direct B, fixed pairs, staged policies, and adaptive selection under the normalized calibrated recipe. The frequency control in Table 23 keeps companion frequencies within each PDE and refits them in every resample. Table 24 compares static T/TB mixtures chosen to match training cost. When the adaptive budget lies outside the two endpoint costs, the mixture uses a boundary probability and is charged that endpoint's actual cost.

The stop-nonB replay continues only T when the selected companion is not B, and continues both when B is selected. Table 25 gives the savings on completed trajectories for which this change preserves the returned state. The completed validation ordering justifies that retrospective comparison; using the rule on a new instance requires evidence that the ordering will persist.

The reported-results archive provides complete N same-run policy levels and intervals in `reported_results/csv/tab_recipeNsame.csv`; the matching source table retains the column labels and units. Other recipe files give absolute errors and saved-state controls, including fixed B and T/B comparisons.

Table 20: Paired recipe transitions on 80 contexts. Counts are changed final retained sets, changed returned actions, and distinct seed clusters containing action changes. The comparisons use the same matched instances.

Rule	Change	Policy	Sets	Actions	Clusters
Same-run	O→R	T+validation	5	0	0
Same-run	O→R	Full set	0	0	0
Same-run	O→R	Top-2 at 100	8	1	1
Same-run	O→R	Stage 100/600	6	6	5
Same-run	O→R	Stage 300/900	0	0	0
Same-run	R→N	T+validation	8	1	1
Same-run	R→N	Full set	0	0	0
Same-run	R→N	Top-2 at 100	7	3	2
Same-run	R→N	Stage 100/600	2	3	2
Same-run	R→N	Stage 300/900	0	0	0
Same-run	O→N	T+validation	11	1	1
Same-run	O→N	Full set	0	0	0
Same-run	O→N	Top-2 at 100	11	3	2
Same-run	O→N	Stage 100/600	8	9	6
Same-run	O→N	Stage 300/900	0	0	0
Cross-initialization	O→R	T+validation	3	0	0
Cross-initialization	O→R	Full set	0	0	0
Cross-initialization	O→R	Top-2 at 100	2	0	0
Cross-initialization	O→R	Stage 100/600	4	4	2
Cross-initialization	O→R	Stage 300/900	0	0	0
Cross-initialization	R→N	T+validation	2	2	2
Cross-initialization	R→N	Full set	0	0	0
Cross-initialization	R→N	Top-2 at 100	9	1	1
Cross-initialization	R→N	Stage 100/600	3	3	2
Cross-initialization	R→N	Stage 300/900	0	0	0
Cross-initialization	O→N	T+validation	5	2	2
Cross-initialization	O→N	Full set	0	0	0
Cross-initialization	O→N	Top-2 at 100	11	1	1
Cross-initialization	O→N	Stage 100/600	5	5	3
Cross-initialization	O→N	Stage 300/900	0	0	0

Table 21: Low-rate extensions minus their preceding calibrated settings. Set and state columns count changes among 80 matched contexts; state equality includes the parameter hash. Gap differences are pp to the baseline common reference, with pointwise 95% intervals. Same and Cross denote same-run and cross-initialization selection.

Comparison	Selection	Policy	Sets	States	Gap difference [95%]
$R^+ - R$	Same	T+validation (T fallback)	9	0	$0.000\ [0.000, 0.000]$
$R^+ - R$	Same	Stage 100/600	3	3	$-0.022\ [-0.122, 0.078]$
$R^+ - R$	Same	Stage 300/900	0	0	$0.000\ [0.000, 0.000]$
$R^+ - R$	Cross	T+validation (T fallback)	2	2	$-0.134\ [-0.402, 0.000]$
$R^+ - R$	Cross	Stage 100/600	2	2	$-0.111\ [-0.274, 0.000]$
$R^+ - R$	Cross	Stage 300/900	0	0	$0.000\ [0.000, 0.000]$
$N^+ - N$	Same	T+validation (T fallback)	8	1	$0.033\ [0.000, 0.100]$
$N^+ - N$	Same	Stage 100/600	5	5	$0.705\ [0.008, 1.967]$
$N^+ - N$	Same	Stage 300/900	0	0	$0.000\ [0.000, 0.000]$
$N^+ - N$	Cross	T+validation (T fallback)	6	4	$0.063\ [-0.180, 0.370]$
$N^+ - N$	Cross	Stage 100/600	3	3	$0.061\ [-0.278, 0.460]$
$N^+ - N$	Cross	Stage 300/900	0	0	$0.000\ [0.000, 0.000]$

Table 22: Same-run policy levels for the normalized calibrated recipe N, on 80 contexts. Gaps use the baseline cross-initialization reference. Work is mean GPU-s. Paired frequency and static comparisons follow in Tables 23–24.

Policy	Mean gap (%)	Mean nRMSE	Work (GPU-s)
Fixed T	2.10	0.4317	994
Fixed B	–0.35	0.4232	1,902
Fixed T/B	–1.93	0.4160	2,896
T+random after	1.30	0.4286	1,870
T+validation	–0.05	0.4230	2,424
Static T–T/B mixture	–0.93	0.4199	2,424
Stage 100/600	1.70	0.4319	1,785
Stage 300/900	–0.87	0.4207	2,243
Full set	–1.93	0.4160	4,848

Table 23: T+validation minus its within-PDE frequency control. Gap and work intervals are paired, pointwise 95%; frequencies are refitted in each draw. Each row has 80 contexts and 16 seed clusters.

Rule	Recipe	Gap difference [95%] (pp)	Work difference [95%] (GPU-s)
Same-run	O	0.257 $[-0.198, 0.703]$	0.6 $[-2.3, 3.7]$
Same-run	R	0.252 $[-0.213, 0.700]$	–0.1 $[-2.6, 2.4]$
Same-run	N	0.455 $[-0.066, 0.976]$	0.1 $[-2.6, 2.8]$
Cross-initialization	O	–0.132 $[-0.479, 0.286]$	3.9 $[-5.2, 12.9]$
Cross-initialization	R	–0.109 $[-0.460, 0.315]$	0.9 $[-4.4, 6.5]$
Cross-initialization	N	–0.106 $[-0.481, 0.341]$	0.8 $[-4.5, 6.3]$

Table 24: T+validation minus a fitted static T–T/B mixture. Mean-gap differences use pointwise 95% intervals. Clipped draws are out of 20,000; those draws use a boundary mixture rather than exact mean-work matching.

Rule	Recipe	p	Gap difference [95%] (pp)	Clipped draws
Same-run	O	0.779	0.775 $[-0.013, 1.572]$	0
Same-run	R	0.770	0.740 $[-0.058, 1.546]$	0
Same-run	N	0.752	0.880 $[0.008, 1.739]$	0
Cross-initialization	O	0.864	0.357 $[-0.175, 1.041]$	257
Cross-initialization	R	0.880	0.407 $[-0.135, 1.094]$	581
Cross-initialization	N	0.869	0.390 $[-0.222, 1.103]$	396

Table 25: Stopping non-B continuation after screening. Savings are T+validation minus the stop policy on the same 80 contexts; all rows preserve 80 returned actions. Intervals are paired 95%; saving percentages divide mean saved work by T+validation mean work.

Recipe	Selection	Same actions	Saving (GPU-s)	95% interval	Saving (%)
O	Same-run	80	153.52	$[72.78, 251.19]$	6.20
O	Cross-initialization	80	438.90	$[51.87, 917.16]$	4.16
R	Same-run	80	161.11	$[84.96, 244.02]$	6.55
R	Cross-initialization	80	285.87	$[49.70, 612.25]$	2.68
N	Same-run	80	173.00	$[98.83, 247.86]$	7.14
N	Cross-initialization	80	386.69	$[89.90, 808.54]$	3.65
R^+	Same-run	80	154.04	$[90.01, 220.03]$	6.11
R^+	Cross-initialization	80	383.26	$[50.04, 900.92]$	3.62
N^+	Same-run	80	90.86	$[50.94, 129.03]$	3.70
N^+	Cross-initialization	80	110.03	$[23.46, 209.56]$	1.02

F Classification Records and Inference

F.1 Source Records, Donor Construction, and Audit Tasks

LCBench provides learning curves for 2,000 neural-network configurations on each of 35 classification datasets (Zimmer et al., 2021). We match configurations using the identifiers shared across these datasets, extract the 50 epochs after initialization, and convert validation and test accuracy to error in percentage points. Subtracting the initialization timestamp gives cumulative recorded time. Policy evaluation then selects from these observed states.

A deterministic metadata-based split assigns ten datasets to donor construction and 25 to audit evaluation, with disjoint source task identifiers. The donor datasets are jasmine, covertype, volkert, dionis, bank-marketing, connect-4, credit-g, phoneme, jungle_chess_2pcs_raw_endgame_complete, and segment. The audit datasets are APSFailure, vehicle, airlines, mfeat-factors, kc1, helena, blood-transfusion-service-center, shuttle, adult, cnae-9, kr-vs-kp, fabert, jannis, Fashion-MNIST, KDDCup09_appetency, higgs, sylvine, Amazon_employee_access, christine, albert, car, numerai28.6, Australian, nomao, and MiniBooNE.

The donor rank averages within-dataset percentile ranks of each configuration's best recorded validation error. Fixed configuration identifiers resolve ties. The fixed LCBench design takes the top 32 configurations and uses the first as the donor-selected default. Audit validation selects the companion at epoch 5; both survivors train to epoch 50. The fixed-design return rule chooses the lowest-validation state over all recorded epochs of those final survivors. Every audit outcome is evaluated on that dataset's separate test record. Configuration identities, states, and source arrays are recorded so that policy actions can be reconstructed.

The archived workflow locks the two comparisons, donor-built pool and default, epoch-5 screening, survivor-best return, and analysis code before computing audit policy outcomes. Its dated lock is from 20 September 2026, 18:35:59 UTC; the audit script checks the locked code before loading the audit metrics. Subsequent return-rule and pool expansions follow exposure to the fixed-design results.

The LCBench learning-curve data are distributed under the Creative Commons Attribution 4.0 International license (CC BY 4.0). LCBench supplies configuration trajectories on classification datasets. Dataset variation is the outer sampling unit in this application; every audit decision uses the dataset's own validation and test records.

F.2 Return Rules and Resource Alternatives

For the fixed 32-candidate, epoch-five design, cap-only searches the two survivor endpoints at epoch fifty; survivor-best searches all their saved epochs; all-reached also admits the computed prefixes of discarded candidates. Every policy uses the same context's trajectories. Table 26 records the three output profiles; Table 27 gives their observed matching and test-informed fixed-frequency opportunities.

Table 26: Fixed-design LCBench return comparison. Errors are percentages; matching is frequency-control error minus adaptive error (pp). Actions are the mean number of distinct outputs across companion routes per audit dataset.

Return rule	Adaptive error	Control error	Matching	Actions
Cap-only	20.233	21.005	0.772	12.24
Survivor-best	18.895	19.145	0.250	8.68
All-reached	18.771	18.959	0.188	4.84

Table 28 retains the survivor-best policy levels, including fixed, random, adaptive, and staged alternatives. Table 29 gives their paired differences with the protocol-defined coverage. Fixed singleton and pair identities come from donor validation; audit test records evaluate the returned predictions. The donor-static linear program chooses probabilities over direct singletons and default pairs using donor validation subject to a mean-time budget on those donor datasets. Table 30 records their audit outcomes at the costs actually incurred.

In the two endpoint mixtures, the adaptive time allowance exceeds the cost of the donor pair. Both mixtures therefore use the pair with probability one, explaining their identical rows in Table 28. The cap-only comparisons and task-level results are retained in the reported-results archive.

Table 27: Fixed-design LCBench matching and its finite audit-loss bound (classification-error pp). Each return rule uses the same 25 datasets and companion frequencies. The optimum allocation uses audit test losses to bound the observed assignment.

Return rule	Observed gain	Maximum gain	Optimal audit error
Cap-only	0.772	1.277	19.727
Survivor-best	0.250	0.799	18.346
All-reached	0.188	0.557	18.402

Table 28: Fixed-design LCBench levels, survivor-best return. Error and q90 are classification-error percentages; recorded time is cumulative recorded seconds. Level intervals are pointwise 95% over 25 audit datasets.

Policy	Mean error	95% interval	q90 error	Recorded time (s)
Fixed default	19.355	$[12.988, 26.659]$	37.622	470.75
Fixed donor-rank 2	19.535	$[13.257, 26.699]$	37.114	411.27
Fixed donor pair	18.952	$[12.591, 26.164]$	37.114	882.01
Full pool	18.463	$[12.176, 25.566]$	36.524	12,770.21
Validation at 5	18.895	$[12.608, 26.076]$	36.524	2,023.07
Validation at 10	18.715	$[12.470, 25.793]$	37.353	3,350.00
Unrestricted Top-2 at 5	19.060	$[12.774, 26.189]$	36.524	1,876.25
Staged pruning	18.812	$[12.541, 25.897]$	36.524	2,323.17
Uniform companion	19.223	$[12.901, 26.441]$	37.622	1,942.19
Audit-frequency control	19.145	$[12.806, 26.358]$	37.622	2,011.82
Donor-frequency control	19.153	$[12.847, 26.366]$	37.342	1,914.09
Default/pair mixture	18.952	$[12.591, 26.164]$	37.114	882.01
Second/pair mixture	18.952	$[12.591, 26.164]$	37.114	882.01

Table 29: Fixed-design survivor-best paired contrasts: validation at epoch 5 minus the listed comparator. Intervals have the recorded coverage; 97.5% rows are the two prespecified primary comparisons. Time differences use recorded seconds; a dash denotes an unreported timing contrast.

Comparator	Difference (pp)	Interval	Coverage	Extra time (s)
Fixed default	-0.460	$[-0.816, -0.150]$	95.0%	1,552.3
Fixed donor-rank 2	-0.640	$[-1.760, 0.199]$	95.0%	1,611.8
Fixed donor pair	-0.057	$[-0.442, 0.365]$	95.0%	1,141.1
Full pool	0.432	$[0.117, 0.815]$	95.0%	-10,747.1
Validation at 10	0.180	$[-0.022, 0.429]$	95.0%	-1,326.9
Unrestricted Top-2 at 5	-0.165	$[-0.593, 0.189]$	95.0%	146.8
Staged pruning	0.083	$[-0.295, 0.447]$	95.0%	-300.1
Uniform companion	-0.328	$[-0.630, -0.062]$	97.5%	80.9
Audit-frequency control	-0.250	$[-0.499, -0.029]$	97.5%	11.3
Donor-frequency control	-0.259	$[-0.480, -0.054]$	95.0%	109.0
Default/pair mixture	-0.057	$[-0.442, 0.365]$	95.0%	1,141.1
Second/pair mixture	-0.057	$[-0.442, 0.365]$	95.0%	1,141.1
Donor-frequency refit	-0.259	$[-0.504, -0.027]$	95.0%	–

Table 30: Adaptive and donor-static policy means at the fixed 32-candidate, epoch-5 design. Errors are percentages and times are recorded seconds. Donor-static probabilities are fit on donor validation and costs, then held fixed on the 25 audit datasets.

Return rule	Adaptive error	Adaptive time	Donor-static error	Donor-static time
All-reached	18.771	2,023.07	19.075	780.55
Survivor-best	18.895	2,023.07	19.075	780.55
Cap-only	20.233	2,023.07	21.069	780.55

F.3 Fixed-Design and Return-Rule Intervals

The fixed-design analysis compares the adaptive policy with uniform and target-frequency companions using the donor-built pool and default. The frequency control's probabilities are estimated from the audit policy and re-estimated in every draw. Table 5 distinguishes this diagnostic from a distribution learned on donor data and then deployed.

The two primary contrasts retain the fixed-design 10,000-draw audit-dataset bootstrap and two-sided 97.5% intervals. Return-rule comparisons are post-outcome paired diagnostics on the same 25 datasets. Table 31 gives both pointwise and four-contrast simultaneous 95% intervals. The simultaneous procedure centers each contrast at its bootstrap mean, divides by its fixed bootstrap standard deviation, and uses the maximum absolute standardized deviation. The standardizing value is fixed across draws for each contrast. In the cap-only to survivor-best comparison, the control improves more on 15 of 25 datasets. The expansion from survivor-best to all-reached favors the control's improvement on five datasets, the adaptive policy on three, and leaves seventeen tied.

Table 31: Post-outcome paired changes between return rules at the fixed design (classification-error pp). Intervals report pointwise and simultaneous 95% uncertainty for the four displayed diagnostics.

New minus old	Quantity	Change (pp)	Pointwise 95%	Simultaneous 95%
Survivor − cap	Adaptive test error	-1.338	$[-2.9200, -0.1660]$	$[-3.199, 0.522]$
Survivor − cap	Matching	-0.522	$[-1.4667, 0.1046]$	$[-1.641, 0.597]$
All − survivor	Adaptive test error	-0.124	$[-0.3557, -0.0002]$	$[-0.399, 0.151]$
All − survivor	Matching	-0.062	$[-0.1886, 0.0610]$	$[-0.226, 0.102]$

F.4 Construction and Broad-Family Sensitivity

The extended analysis evaluates 3,765 conditions from 247 pools, varying pool construction, candidate count, screening budget, donor-default rank, and return rule. It summarizes 345 groups on the same audit datasets. For sampled-pool families, pool draws are averaged within an audit dataset before dataset-level inference. The 540-comparison simultaneous family includes two adaptive contrasts across three pool families, six pool sizes, five budgets, and three return rules. All its max-standardized intervals contain zero.

Full-pipeline refitting resamples donor and audit datasets independently. Each draw rebuilds the ranking, pool, default, companion frequencies, and donor-static probabilities, then evaluates those choices on the resampled audit records. There are 2,000 draws per setting. Table 32 reports the resulting intervals, including $[-0.017, 1.303]$ pp for matching at the fixed-design settings. Its point summaries are means over the refitted procedures; Table 3 retains the estimates for the fixed pool and default.

F.5 Validation Observations and Recorded Time

Recorded LCBench time is cumulative source runtime, including its evaluation schedule. The three return rules share the training stops, while required validation observations number 34, 130, and 250.

Table 32: Full donor/audit refit sensitivity under survivor-best return. Each row uses 2,000 pipeline-refit draws at the indicated pool size and screening epoch; error differences are pp. These intervals incorporate donor construction as well as audit sampling.

Pool size	Screen epoch	Contrast	97.5% refit interval (pp)
32	5	Validation − uniform	$[-1.774, -0.056]$
32	5	Validation − audit frequency	$[-1.303, 0.017]$
32	5	Validation − donor frequency	$[-1.407, -0.003]$
32	5	Validation − donor static	$[-1.238, 0.453]$
32	10	Validation − uniform	$[-1.604, -0.096]$
32	10	Validation − audit frequency	$[-1.067, -0.020]$
32	10	Validation − donor frequency	$[-1.173, 0.060]$
32	10	Validation − donor static	$[-1.157, 0.626]$
128	5	Validation − uniform	$[-2.732, -0.111]$
128	5	Validation − audit frequency	$[-1.899, 0.069]$
128	5	Validation − donor frequency	$[-2.559, 0.063]$
128	5	Validation − donor static	$[-1.550, 0.370]$
128	10	Validation − uniform	$[-2.716, -0.057]$
128	10	Validation − audit frequency	$[-1.986, 0.061]$
128	10	Validation − donor frequency	$[-2.307, 0.126]$
128	10	Validation − donor static	$[-1.503, 0.386]$

G Training and Calibrated Validation Costs

G.1 Executed Training

PDE training costs sum exclusive-device intervals for every candidate at its actual stopping update. Final survivors are charged through the cap, even when final validation chooses an earlier saved state. Cross-initialization charges the four selection runs and excludes later training of the evaluated held-out run, consistently with the report. Same-run charges one run. The active-candidate validation schedule below adds the score requests required by that implementation; the measured training-only axes in the figures remain unchanged.

The reported adaptation cost assumes that the default and recipe have already been chosen. Acquiring the historical evidence and calibrating the recipe incur separate costs; a reused run is counted once. Training and validation totals exclude model construction and loading, disk reads, parameter hashing, checkpoint management, and research test evaluation.

G.2 Validation-Aware Route Costs

Measured Operation and Coverage. A complete validation call evaluates four sequences with the production helper, using FP32, one sequence at a time, and a 15-step autoregressive rollout. Host timers bracket the helper with CUDA synchronization. They include its transfers, metric aggregation, and finite checks. Measurements use loaded models and host-resident arrays on exclusively assigned RTX 4090 D devices, with one measurement worker per physical host. The recorded software is PyTorch 2.5.1 and CUDA 12.1.

For each model–task cell, the calibration samples three dataset seeds, two actual host sessions per seed, and five measured calls after two warm-ups. We first average calls within session, then actual hosts within seed, and finally seeds equally. The main measurements use the initialized task interface and pretrained parameters before adaptation. A different initialization for one seed per cell supplies a sensitivity check, not an additional seed in the main mean. Both parameter and random-state hashes are unchanged by the calls, and no test sequences are accessed.

Validation calibration covers every model–task cell in the external PDE study (Table 33); validation-aware costs are reported for that study, while the six-task report uses measured training costs.

Table 33: Complete-call validation calibration for the external PDE study, in seconds. Each main rate averages five calls per session, two actual hosts per seed, and three seeds. One call evaluates all four validation sequences. The models are in their initialized pre-adaptation states.

Model	MHD	CGL
Poseidon-T	5.069	5.020
Poseidon-B	9.456	9.517
MPP-Ti	3.885	3.853
MPP-L	7.670	7.659
MORPH-Ti	0.729	0.731
MORPH-M	1.357	1.358

Requests Needed to Implement a Route. The calibrated execution evaluates every legal saved state of a still-active candidate as training reaches it. Scores are retained for final survivor-best selection, and repeated uses of a state are charged once. Let n_{ckm} count these calls to model m under route k for context c, including the four selection runs for cross-initialization. With task $p(c)$ and calibrated call duration $\widehat{v}_{m,p(c)}$, reconstructed work is

$$\widehat{C}^{\mathrm{tv}}_{ck} = C^{\mathrm{train}}_{ck} + \sum_{m} n_{ckm}\widehat{v}_{m,p(c)}. \tag{5}$$

The first term is the measured training time. The second multiplies the number of required validation calls by the calibrated duration of each call. Equation 5 sums them for the active-candidate schedule, using the selection-run accounting and cost scope defined in Appendix G.

Table 34 gives the active-candidate validation counts used in the reconstructed costs. For Stage 300/900, evaluating each still-active candidate at its recorded checkpoints requires $6 \times 3 + 3 \times 3 + 1 \times 5 = 32$ calls per run.

Table 34: Active-candidate validation requests per initialization under survivor-best return. Adaptive companion and Top-2 at 100 share one schedule: all six candidates are scored at update 100 and two continue. These counts enter the calibrated total; multiply by four for cross-initialization selection.

Policy	Validation requests
Fixed T	11
Fixed B	11
Fixed T/B	22
Adaptive companion	26
Top-2 at 100	26
Top-1 at 600	36
Top-1 at 900	41
Stage 100/600	24
Stage 300/900	32
Full set	66

Comparing Routes with the Same Returned Predictor. All external policy results use the same 120 contexts, 24 dataset-seed clusters, reference denominators, and recorded actions as the training-only evaluation. Random routes are reconstructed member by member before taking their finite conditional expectation. Table 35 adds calibrated validation to the training-only comparisons in Table 2 and the corresponding figures.

Fixed T/B and the full pool select identical models, updates, parameter hashes, and test outcomes in all 120 contexts under each selection rule. Their cross-initialization totals are 12,211 and 20,684 seconds, a 40.96% saving; same-run totals have the same ratio. The equality of returned states makes this a comparison of the resources used to obtain the same recorded predictions. Fixed B and the adaptive companion policy instead have different returned states in some contexts, so their quality comparison remains part of that operating choice. Including calibrated validation gives 8,016 versus 10,330 seconds under cross-initialization, or 22.40% less work for B.

Stage 300/900 also returns the same action as Top-1 at 900 in all 120 cross-initialization contexts. It saves 1,308 seconds of training and 136 seconds of validation, totaling 1,445 seconds or 12.87%.

Their same-run actions agree in 114 contexts, so that selection rule retains a predictive comparison as well as a cost difference. These additional schedules use the same recorded trajectories and the diagnostic role in Table 5.

Table 35: External policy costs in device seconds. Training, validation, and total sum the four cross-initialization selection runs; the same-run total charges only the evaluated run under same-run selection. Training is measured, validation uses Table 33, and totals follow Equation 5. Random rows average their finite members.

Policy	Training	Validation	Total	Same-run total
Fixed T	3,973	222	4,195	1,049
Fixed B	7,599	417	8,016	2,004
Fixed T/B	11,572	639	12,211	3,053
Adaptive companion	9,731	599	10,330	2,501
T+random before screening	7,067	425	7,493	1,873
T+random after screening	7,493	499	7,992	1,998
Uniform pair after screening	6,927	488	7,415	1,854
Top-1 at 100	4,975	348	5,323	1,378
Top-2 at 100	9,568	604	10,172	2,390
Top-1 at 600	8,514	736	9,250	2,294
Top-1 at 900	10,385	838	11,222	2,784
Stage 100/600	7,628	561	8,190	1,892
Stage 300/900	9,076	701	9,778	2,383
Largest singleton	4,182	337	4,519	1,130
Largest pair	11,781	755	12,536	3,134
Full set	19,446	1,239	20,684	5,171

Sampling and Calibration Sensitivity. For the planned B-versus-adaptive comparisons under the two selection rules, the timing protocol specifies two-sided 97.5% intervals. Additional cost contrasts use 95% intervals. All use 20,000 shared resamples of dataset seeds within each PDE, carrying the five initialization views and paired policy outcomes together. The calibrated model–task rates are held fixed in this conditional analysis (Table 36). Timing repeats enter the cell calibration; prediction uncertainty is measured across dataset seeds.

Validation calls from 144 development runs, with fifteen calls during adaptation, support the calibrated rate scale: in each external model–task cell, the initialized-state calibration mean differs from the training-phase mean by at most 2.21%. Varying the twelve shared rates across their observed session/development ranges produces 4,096 corner combinations, with each chosen rate used for every policy call to its model and task. The resulting B-versus-adaptive saving ranges from 22.26% to 22.52%; using development-run mean rates gives 22.37%.

Table 36: Paired cost comparisons conditional on the calibrated rates. Saving is $100(1-C_{\text{left}}/C_{\text{right}})$; intervals use the same clustered resamples for both policies. The two planned B comparisons have 97.5% coverage; other displayed intervals use 95% coverage. Cross and Same mean cross-initialization and same-run selection; the "Same" column counts identical returned identity, update, and hash among 120 contexts.

Rule	Left / right	Same	Saving (%)	Coverage	Interval (%)
Cross	B / adaptive	58	22.40	97.5%	[11.31, 30.37]
Cross	T/B / full	120	40.96	95%	[40.85, 41.10]
Cross	Stage / Top-1 at 900	120	12.87	95%	[12.27, 13.50]
Same	B / adaptive	37	19.86	97.5%	[13.50, 25.41]
Same	T/B / full	120	40.96	95%	[40.85, 41.10]
Same	Stage / Top-1 at 900	114	14.40	95%	[12.88, 16.19]

The cost axis determines a cost-matched mixture's fitted probabilities and expected loss, while its two endpoint predictors remain unchanged.

H Data, Candidate Models, and Adaptation Settings

The four-system and six-task PDE studies use six PDEs covering phase/reaction, fluid/transport, and wave/dispersive evolution under a shared tensor interface. Each has 480 decision contexts in 48 dataset-seed clusters; the four- and six-system studies contain 1,920 and 2,880 training trajectories, respectively. Each source simulation has 100 saved frames with two channels on a periodic 64×64 grid over $[0, 2\pi)^2$. A deterministic assignment selects one of five non-overlapping 20-frame windows from each sequence. The selected window's first five frames provide the forecast history and its next fifteen provide the prediction targets.

Figure 10 shows the candidate scores available along the six-task adaptation trajectories. Screening uses the recorded score at that update; later scores enter final state selection when the route has computed them.

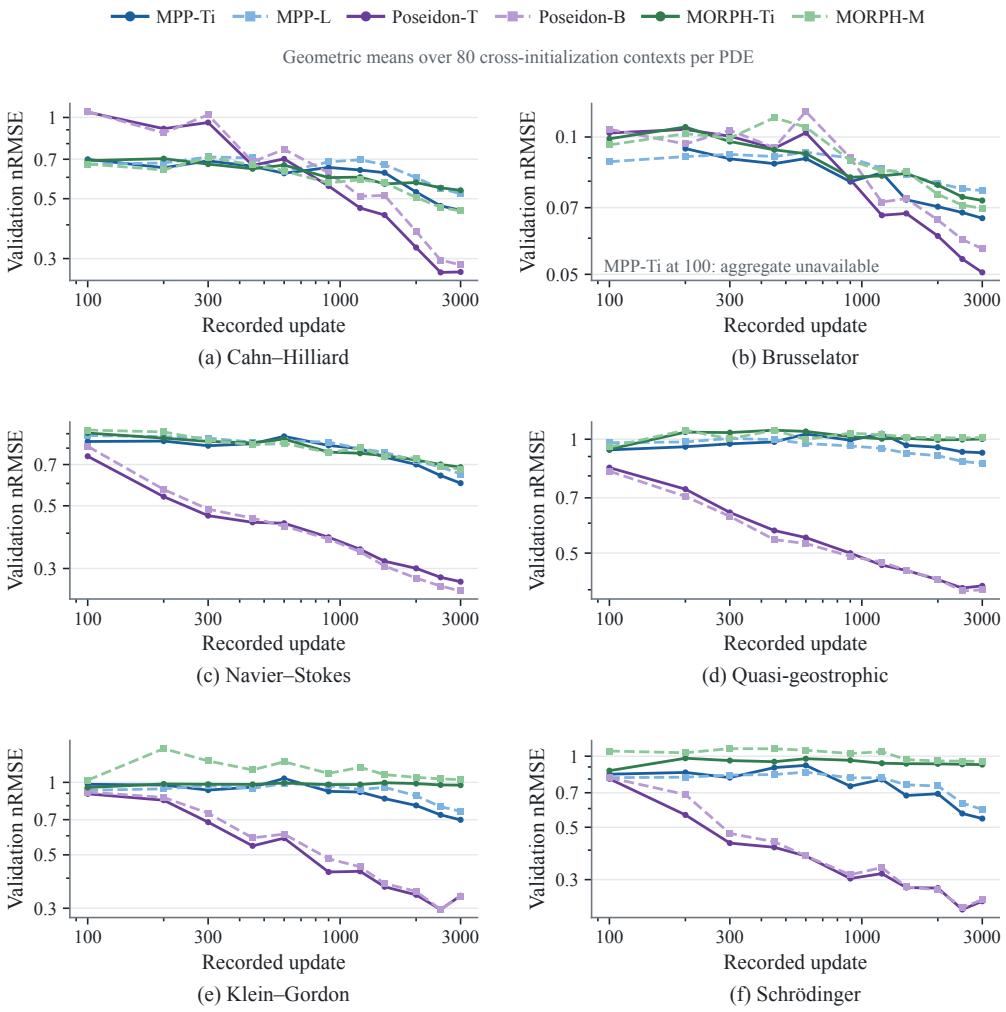


Figure 10: Recorded validation trajectories for the six candidate systems on the six PDE tasks. Curves summarize adaptation scores over saved updates. Early selection at update 100 uses the scores then available; final validation can inspect the eligible states of retained candidates.

At update 100, 22 of 80 MPP-Ti cross-initialization validation scores on Brusselator are non-finite. The plotted geometric mean therefore has no finite value at that point. The recorded states remain present; the finite-score selection rule in Appendix C excludes those scores from that early ranking and retains eligible later states for final validation.

H.1 Physical-Field Examples

Figure 11 displays the first channel of one adaptation window per task. Using zero-based indices, frame 4 closes the five-frame history and frame 19 closes the 15-step forecast target. The centerline views show the entire window. Each solver-generated example is sequence zero of the smallest dataset seed in the six-task PDE study, selected without consulting model outcomes.

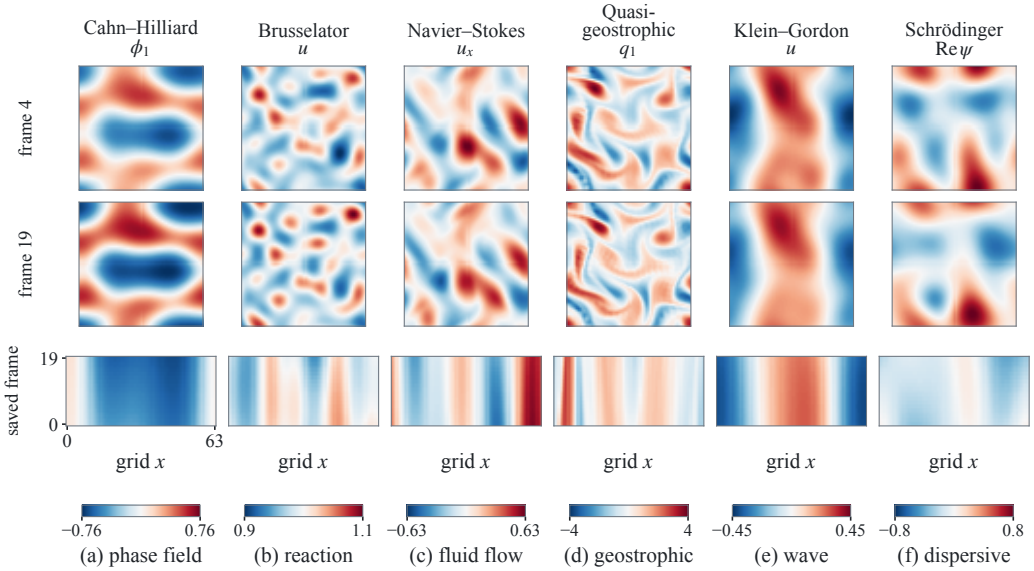


Figure 11: Actual adaptation fields for the six tasks. The first two rows show saved frames 4 and 19 (zero-based); the third shows all 20 frames at grid-row index 32. Each column shares one color scale across its views, symmetric about zero for signed fields and spanning the displayed concentration range for Brusselator. Color bars report native solver values.

H.2 Equations and Numerical Representations

H.2.1 Coupled Cahn–Hilliard

The two phase fields evolve according to

$$\partial_t \phi_i = M_i \Delta\left(-\kappa_i \Delta \phi_i + a_i \phi_i + \phi_i^3 - \rho \phi_{3-i}\right), \qquad i \in \{1, 2\}.$$

The implementation uses a dealiased Fourier pseudospectral discretization and stabilized semi-implicit Euler steps, treating reciprocal coupling and fourth-order terms implicitly. Table 37 gives the jointly indexed dynamical coefficients. Initial spectral bandwidth spans 2–6 and initial RMS amplitude spans 0.08–0.16, with small independently varied field means. Each initial condition is sampled independently under its specified spectral and amplitude setting. The solver step is 0.002 and a frame is saved every ten steps.

Table 37: Coupled Cahn–Hilliard dynamical settings. The same zero-based level indexes all coefficients in a row; $M_1 = 1$ throughout.

Level	a_1	a_2	κ_1	κ_2	ρ	M_2
0	−0.890	−0.760	0.140	0.1850	0.120	0.620
1	−0.925	−0.790	0.130	0.1725	0.145	0.670
2	−0.960	−0.820	0.120	0.1600	0.170	0.720
3	−0.995	−0.850	0.110	0.1475	0.195	0.770
4	−1.030	−0.880	0.100	0.1350	0.220	0.820

H.2.2 BRUSSELATOR

The state comprises two reacting species:

$$\partial_t u = D_u \Delta u + A - (B+1)u + u^2 v, \qquad \partial_t v = D_v \Delta v + Bu - u^2 v.$$

We use second-order Strang splitting, exact Fourier diffusion, and an explicit midpoint reaction flow. Across the dynamical settings, $A = 1$, $B \in \{1.78, 1.815, 1.85, 1.885, 1.92\}$, $D_u \in \{0.01656, 0.01728, 0.018, 0.01872, 0.01944\}$, and $D_v \in \{0.1908, 0.1854, 0.18, 0.1746, 0.1692\}$. Initial perturbations are centered at $(A, B/A)$, with bandwidth 2–6 and RMS amplitude 0.025–0.075. The solver step is 0.02; saved frames are eight steps apart.

H.2.3 DECAYING NAVIER–STOKES

The observable is the two-component incompressible velocity field:

$$\partial_t \mathbf{u} + (\mathbf{u} \cdot \nabla)\mathbf{u} = -\nabla p + \nu \Delta \mathbf{u}, \qquad \nabla \cdot \mathbf{u} = 0.$$

There is no forcing. The implementation evolves vorticity with Fourier pseudospectral RK4 and two-thirds dealiasing, then obtains velocity through modal inversion. Viscosities are $\{0.003, 0.0025, 0.002, 0.0015, 0.001\}$. Initial spectral centers are $\{2.0, 2.8, 3.6, 4.4, 5.2\}$, and initial speed RMS values are $\{0.30, 0.35, 0.40, 0.45, 0.50\}$. A solver step of 0.02 and stride five give a saved interval of 0.1.

H.2.4 TWO-LAYER QUASI-GEOSTROPHIC DYNAMICS

The two channels are potential-vorticity anomalies, with equal layer depths and reciprocal coupling:

$$q_1 = \Delta\psi_1 + F(\psi_2 - \psi_1), \qquad q_2 = \Delta\psi_2 + F(\psi_1 - \psi_2),$$
$$\partial_t q_i + J(\psi_i, q_i) + \beta \partial_x \psi_i = \nu \Delta q_i - \mathbf{1}\{i = 2\}\, r \Delta \psi_2.$$

Here $F = 8$, and $J(f, g) = f_x g_y - f_y g_x$. The unforced periodic beta-plane anomaly equations are advanced by pseudospectral RK4 with exact modal inversion and two-thirds dealiasing. The five settings use $\beta = \{0.2, 0.35, 0.5, 0.65, 0.8\}$, bottom drag $r = \{0.06, 0.05, 0.04, 0.03, 0.02\}$, and viscosity $\{0.0015, 0.0012, 0.001, 0.0008, 0.0006\}$. Initial spectral centers span 1.8–4.6 and layer-velocity RMS spans 0.18–0.34. Frames are saved every five steps of size 0.02.

H.2.5 NONLINEAR KLEIN–GORDON

The state is $(u, \partial_t u)$, with

$$\partial_{tt} u = c^2 \Delta u - m^2 u - \lambda u^3.$$

The integrator alternates a nonlinear kick with the exact Fourier linear Klein–Gordon flow using symmetric second-order splitting. The five dynamical settings are $c = \{0.70, 0.82, 0.94, 1.06, 1.18\}$, $m = \{0.8, 0.9, 1.0, 1.1, 1.2\}$, and $\lambda = \{0.3, 0.5, 0.7, 0.9, 1.1\}$. Initial bandwidth ranges from 2 to 6, displacement RMS from 0.18 to 0.34, and velocity RMS from 0.07 to 0.15. The solver step is 0.005, with saved stride four.

H.2.6 NONLINEAR SCHRÖDINGER

The real and imaginary parts of ψ form the two-channel state:

$$\mathrm{i}\partial_t \psi = -\kappa \Delta \psi + g|\psi|^2 \psi.$$

The cubic nonlinearity is defocusing. We use second-order Strang splitting between exact Fourier linear evolution and the pointwise cubic flow. The five settings use $\kappa = \{0.35, 0.45, 0.55, 0.65, 0.75\}$ and $g = \{0.5, 0.75, 1.0, 1.25, 1.5\}$. Initial bandwidth spans 2–6 and field RMS spans 0.34–0.66. The step is 0.0025 and saved stride eight.

H.3 DATA ROLES AND NUMERICAL CHECKS

For each PDE and dataset seed, adaptation, validation, and test sequences are generated independently. Adaptation uses eight combinations of intermediate initial-condition and dynamical complexity. Test uses the full five-by-five grid of ordered complexity levels, with four sequences per cell. This produces 100 test sequences per dataset seed, including settings outside the intermediate adaptation levels. The

five initialization runs use the same arrays; uncertainty due to drawing a new dataset is represented by the dataset-seed clusters.

The full-window numerical check covers all 10,752 unique source sequences across the four-system and six-task PDE studies (5,376 each), including adaptation, validation, and test. Reconstructing each selected window reproduces its cached array byte for byte. Supervision regimes, candidate systems, and initialization runs reuse these arrays.

For each source sequence, temporal refinement compares step sizes Δt and $\Delta t/2$ in two runs: from the source initial condition through the selected window, and from the cached state at the start of that window. Both comparisons use identical physical saved times. The diagnostic is the largest joint-channel relative RMS difference over the fifteen forecast frames, normalized by the finer solution's RMS. The largest same-initial-condition discrepancy is 0.02298 for Cahn–Hilliard; Brusselator reaches 0.002617 and the other tasks remain below 2.7×10^{-5}. Cahn–Hilliard's largest window-restart discrepancy is 0.01524.

Quarter-step refinement was evaluated for 47 selected source sequences, including the largest-error sequence from every Cahn–Hilliard dataset seed. For Cahn–Hilliard, the median ratio of full/half-step discrepancy to half/quarter-step discrepancy is approximately 1.97. The largest Cahn–Hilliard half/quarter-step discrepancy is 0.01169 from the source initial condition and 0.00770 after restarting at the window boundary.

Task definitions and simulation distributions determine what neural PDE evaluation measures (Li et al., 2021; Musekamp et al., 2025). The four-system and six-task PDE studies share the equations and numerical generation rules but use separate data and initialization seeds. The six-task PDE study covers the stated dynamics under a common interface. Scientific surrogate datasets also span heterogeneous simulation and experimental conditions (Ohana et al., 2024; Hu et al., 2026); rollout-focused methods study error accumulation and frequency content (Lippe et al., 2023). Here the generated fields define the prediction target for the policy comparisons. Appendices H.6 and H.7 specify the external generators and numerical checks.

H.4 Rollout Error Metrics

For an evaluation split and a returned state, let $\mathcal{I}$ index all evaluated sequence, forecast-step, channel, and spatial entries. With predictions $\hat{y}_\ell$ and targets y_ℓ, the implementation computes

$$\mathrm{nRMSE}_{\mathcal{I}} = \sqrt{\frac{\sum_{\ell\in\mathcal{I}}(\hat{y}_\ell - y_\ell)^2}{\max\{\sum_{\ell\in\mathcal{I}} y_\ell^2, 10^{-30}\}}}, \qquad \mathrm{MAE}_{\mathcal{I}} = \frac{1}{|\mathcal{I}|}\sum_{\ell\in\mathcal{I}} |\hat{y}_\ell - y_\ell|.$$

Squared errors and target energies are accumulated in float64 before normalization. The numerical floor protects a zero-energy denominator; the ratios and log scores used for policy selection require positive finite errors. Pooling squared errors and energies differs from averaging separately normalized sequence or pixel errors.

The principal metric uses all 15 predicted steps. Restricting $\mathcal{I}$ to the first ten defines the in-horizon metric; restricting it to the final five defines the late-rollout metric. Both use the same selected predictor. Absolute selected-policy nRMSE appears in Appendix D; Appendix C defines the per-context relative gaps.

H.5 Pretrained Systems and Adaptation Recipes

H.5.1 Systems and Adaptation Schedule

Table 38 lists the instantiated parameter counts.

MPP, Poseidon, and MORPH provide distinct pretrained representations and interfaces (McCabe et al., 2024; Herde et al., 2024; Rautela et al., 2025). Their common output shape permits evaluation on the same target fields while retaining architecture-specific processing. MPP consumes the five-frame history; Poseidon and MORPH use the latest frame through their time-conditioned or autoregressive interfaces. Architecture, pretraining data, parameter count, and adapters are components of each compared system.

For the default-preserving reports, all PDE candidates train through update 100 before the policy keeps T and one companion; fixed singletons and T/B train their specified candidates directly.

Table 38: Trainable parameter counts in the six-task PDE implementation. Counts include the instantiated task interface. "Small" and "large" identify variants within each family.

Family	System	Trainable parameters
MPP	AViT-Ti	7,287,518
	AViT-L	407,064,334
Poseidon	T	20,771,070
	B	157,723,542
MORPH	Ti	9,864,408
	M	125,602,008

Unrestricted Top-k retains the early leaders, while Stage 100/600 and Stage 300/900 prune $6 \to 3 \to 1$ at the named updates. All final survivors follow the same cap-dependent schedule through update 3000.

Each adaptation sequence supplies ten five-frame-to-next-frame windows: zero-based starts 0–9 use history frames j_{win} through $j_{\text{win}} + 4$ and target frame $j_{\text{win}} + 5$, where j_{win} is the window start. The eight sequences yield 80 microbatches ordered first by sequence and then by start frame, visited in a fixed cyclic order for 3000 updates at batch size one. The optimizer is AdamW with zero weight decay. All parameters are adapted; named task input/output parameters form a separate group where present. The input/output-to-trunk learning-rate ratio is ten for Poseidon and one for MPP and MORPH, inherited from the fixed adaptation setup. A cosine multiplier takes the form

$$0.1 + 0.9\,\frac{1 + \cos(\pi t/3000)}{2}.$$

The learning rate has no warmup; the multiplier is updated after each optimizer step for the following step. Every saved checkpoint follows the fixed 3000-update schedule. Training uses no gradient clipping.

The four-system and six-task PDE studies select learning rates using independent development data. For each model–PDE pair, multipliers $0.5, 1, 2$ are compared over two development dataset/initialization pairs and both supervision regimes. Each candidate rate is scored by the mean, across those contexts, of the lowest recorded log validation nRMSE. The four existing systems carry their selected rates into the six-task PDE study; calibration of the two added systems uses 144 trajectories. Table 39 reports the rates fixed before its confirmation outcomes. External-task calibration is specified in Appendix H.9 and Table 40.

The clean loss uses all target entries. For sparse noisy supervision, a fixed mask observes 25% of entries and independent additive Gaussian noise has standard deviation 5% of the training channel scale. The mask and noise remain fixed for a given dataset/regime identity. Validation and test evaluation always use clean full fields. Poseidon and MORPH use training-derived external channel normalization in the baseline loss computation; MPP uses its native internal normalization with the physical-space raw MSE objective before the loss intervention.

Recorded observation times are 20, 40, 60, 80, 100, 200, 300, 450, 600, 900, 1200, 1500, 2000, 2500, and 3000 updates. The first four support early extrapolation; the action set begins at 100. Validation and test metrics at an eligible milestone come from the same model state. A non-finite validation score makes that action ineligible under the rules in Appendix C; all main selected-policy test outcomes are finite.

H.5.2 Checkpoints and Execution Settings

The MPP systems load the released AViT-Ti and AViT-L weights and append the two target fields to the pretrained field interface. Poseidon-T and Poseidon-B load released weights with two-channel task input/output layers. The loaded checkpoints use a 128×128 internal grid: Fourier zero-padding upsamples the latest 64×64 input frame, and Fourier-band truncation returns the prediction to 64×64. The transform uses forward-normalized FFTs; the downsampled band is $-32 \le k_x, k_y \le 31$. MORPH uses the released Ti and M foundation-model checkpoints, saved at epochs 225 and 290 respectively, with a two-scalar-field interface. The provenance records identify the checkpoint files, their content hashes, and the source revisions used by each adapter.

The six-task PDE study and paired MPP recipe comparisons use Python 3.11.15, PyTorch 2.5.1+cu121, and CUDA 12.1. Training inputs are float32, and forward/backward updates run

with automatic mixed precision disabled. Evaluation accumulates squared errors and target energies in float64. Appendix G specifies hardware and synchronized training intervals. Reproducing the trajectories requires pretrained weights, generated sequence arrays, and the recorded adapter and optimizer configurations. The simulations contain no human-subject data; upstream weights and code retain their respective licenses.

H.5.3 Relation to Native Training Protocols

Task-specific finetuning is also the evaluation object in Train-before-Test's language-model comparisons (Zhang et al., 2026). Here each PDE candidate is evaluated with its stated adaptation recipe, so the comparison includes both the pretrained system and the adaptation procedure.

The shared target-task budget defines a reproducible system–recipe comparison. Native protocols provide relevant alternatives: Poseidon reports 200-epoch downstream finetuning with model-specific parameter groups, batch size, and learning rates (Herde et al., 2024); MPP normalizes its objective by target energy (McCabe et al., 2024); and MORPH also considers low-rank adaptation for larger variants (Rautela et al., 2025). The paired comparisons evaluate raw and normalized MPP objectives with separately calibrated learning rates, while holding the other training and evaluation settings fixed.

Table 39: Fixed trunk/input–output learning rates, in units of 10^{-4}. CH, BR, NS, QG, KG, and NLS denote the PDEs in their order in Appendix H. Each entry is trunk/I–O; these values were selected by development validation.

System	CH	BR	NS	QG	KG	NLS
MPP-Ti	20/20	10/10	20/20	10/10	10/10	10/10
MPP-L	10/10	10/10	10/10	10/10	10/10	10/10
Poseidon-T	1.2/12	0.6/6	2.4/24	2.4/24	1.2/12	2.4/24
Poseidon-B	2/20	2/20	4/40	4/40	2/20	4/40
MORPH-Ti	2/2	2/2	2/2	0.5/0.5	0.5/0.5	0.5/0.5
MORPH-M	1/1	2/2	2/2	0.5/0.5	2/2	1/1

H.6 External Task Generators

Incompressible Magnetohydrodynamics. On the periodic domain $[0, 2\pi)^2$, the observed state contains vorticity ω and magnetic potential a. With $\Delta\psi = \omega$, $j = \Delta a$, and $J(f, g) = f_x g_y - f_y g_x$, the generator evolves

$$\partial_t \omega = -J(\psi, \omega) + J(a, j) + \nu\Delta\omega, \qquad \partial_t a = -J(\psi, a) + \eta\Delta a.$$

The zero Fourier mode fixes the Poisson inversion convention. Five ordered parameter settings use $\nu = \eta \in \{0.008, 0.006, 0.004, 0.003, 0.002\}$. Initial spectral bandwidth spans 2–6, with velocity and magnetic-field RMS amplitudes 0.3. The production solver uses dealiased Fourier RK4 on a 256×256 grid, step 0.001, and saved interval 0.02. Prescribed low-band 64×64 initial fields are spectrally interpolated to the solver grid. Saved fields are projected to $|k_x|, |k_y| \leq 31$ and restricted to the common 64×64 model interface. The projected field is the common prediction target for all candidate systems.

Complex Ginzburg–Landau Dynamics. The two channels are the real and imaginary parts of z, with

$$\partial_t z = 0.1(1 + \mathrm{i}\alpha)\Delta z + z - (1 + \mathrm{i}\beta)|z|^2 z.$$

Here $\alpha = 1$, and the five parameter settings use $\beta \in \{-0.5, -1, -1.5, -2, -2.5\}$. Initial bandwidth spans 2–6 and RMS amplitude is 0.70. The 64×64 generator uses Strang splitting with exact linear and reaction subflows, step 0.005, and saved interval 0.1. The saved 64×64 collocation field is the prediction target.

Both generators save 100 frames per source trajectory. A deterministic request identity selects one of five nonoverlapping 20-frame windows. The first five frames supply history and the next fifteen supply forecast targets. Each confirmation dataset seed uses eight adaptation, four validation, and 100 test sequences. Test sequences cover five initial-condition complexities crossed with five dynamical settings, four sequences per cell. Development, qualification, and confirmation requests use disjoint seed namespaces.

H.7 Numerical Qualification Before Model Evaluation

Numerical qualification preceded model evaluation and used 100 fixed cases: two tasks, 25 complexity combinations, and two numerical seeds. Four corner cases per task also checked spatial resolution. MHD comparisons used grids through 512×512 with the same physical parameters, initial seeds, and horizon. The production choice was the 256×256 solver with the 64×64 observation defined above.

For the 256/512 MHD comparison, the largest relative-RMS discrepancy in the projected 64-grid observation was 0.00101547, while the corresponding unfiltered fine-field discrepancy was 0.02356693. CGL retained its 64-grid solver after the prescribed temporal and spatial checks. These measurements characterize refinement sensitivity for the defined observations and qualification requests.

The per-run records specify the refinement protocol, request identities, solver versions, and diagnostics. Production caches identify the source sequence and observation operator. All systems train and evaluate on the same observed arrays within each task.

H.8 Model Families and Transfer Settings

Scientific pretraining aims to transfer reusable structure across physical systems (Subramanian et al., 2023). DPOT explores this goal through Fourier-attention denoising over multiple PDE datasets (Hao et al., 2024). The present menu fixes MPP, Poseidon, and MORPH checkpoints and their adaptation interfaces; the audit comparisons hold those candidates and recipes constant. Pretraining diversity is background for choosing the menu, while each policy's measured outcome comes from the stated target adaptation.

H.9 External Adaptation Settings

Published pretraining descriptions list fluid and diffusion–reaction data for MPP, Euler and incompressible Navier–Stokes operators for Poseidon, and three-dimensional MHD among MORPH's tasks (McCabe et al., 2024; Herde et al., 2024; Rautela et al., 2025). These are the documented pretraining sources for the compared systems. The external PDE study's separation criterion concerns the task data used to select the default identities.

Rate calibration preserves the fixed T and T/B defaults (Appendix D.2). Two independent development dataset/initialization pairs per task evaluate three rate multipliers for six models, giving 72 trajectories without test arrays. Table 40 gives the selected rates. Confirmation uses batch size one, 3000 updates, the cap-dependent cosine schedule, fixed data-window order, and the same candidate-specific interfaces as the four-system and six-task PDE studies.

External MPP minimizes raw physical-space MSE; Poseidon and MORPH normalize channels from training data. Appendix E gives the paired MPP objective comparisons.

Table 40: External development-selected trunk/input–output rates, in units of 10^{-4}. Two data/initialization pairs per task evaluate three multipliers before confirmation.

System	MHD	CGL
MPP-Ti	10/10	20/20
MPP-L	10/10	10/10
Poseidon-T	2.4/24	1.2/12
Poseidon-B	4/40	2/20
MORPH-Ti	1/1	0.5/0.5
MORPH-M	0.5/0.5	0.5/0.5