AGENTIFYING AI EVALUATION: WHAT TO EVALUATE, WHEN TO STOP, AND HOW TO ESTIMATE

Anonymous authors
Paper under double-blind review

ABSTRACT

Evaluating modern AI agents is increasingly costly and time-consuming, as benchmarking tasks can require long sequences of model calls and tool interactions. Can we reduce evaluation time and token cost without sacrificing model evaluation accuracy? Recent methods save cost either by selecting a subset of tasks to run or by stopping trajectories early based on partial execution histories, but generally trade lower cost for reduced evaluation accuracy. We introduce an agentic evaluation framework that jointly and adaptively learns what to evaluate and when to stop, with a tailored estimator that preserves evaluation accuracy. The system jointly learns what tasks to evaluate and when to stop execution, while probabilistically randomizing and adapting these decisions to enable statistical correction. By "learning what tasks to evaluate," it uses historical evaluation traces to identify which benchmark tasks are most informative for estimating agent performance. By "deciding when to stop," it actively monitors each trajectory and probabilistically continues only when further execution is expected to provide sufficient additional information about the eventual outcome. We pair this agentic system with a tailored estimator that corrects for adaptive task selection and stopping. Our theoretical analysis shows that the resulting framework preserves model evaluation accuracy while improving precision for a given evaluation budget. Experiments on benchmarking data show that our approach reduces both evaluation time and token cost while preserving accurate model evaluation results.

1 INTRODUCTION

Evaluating modern AI agents can be expensive because each benchmark task may require a long sequence of model calls, tool use, environment interactions, and verification before the final outcome is observed. The cost can accumulate quickly even for a single model: SWE-bench contains $2{,}294$ real-world software engineering tasks (Jimenez et al., 2024), while SWE-agent reports roughly 1.5 minutes of execution per task under its evaluation setup (Yang et al., 2024), which would amount to more than 50 hours of sequential execution for one pass through the full benchmark. This burden grows further when evaluations are repeated across models, model versions, or experimental conditions.

In fact, recent empirical evidence suggests that some benchmark evaluations may provide little additional information about a target agent's performance. First, some task outcomes may already be reasonably predictable from information available before execution, which raises the possibility that evaluation effort could be concentrated on tasks whose outcomes remain uncertain (Polo et al., 2024; Li et al., 2024; Saranathan et al., 2025). Second, for an initiated task, the partial trajectory may sometimes provide enough information to predict the eventual outcome before the run is complete (Shi et al., 2026b; Ruan et al., 2026; Lin et al., 2026); for example, the agent may have clearly gone off track while substantial execution cost still remains. In such cases, continuing the trajectory may provide relatively little additional information.

Exploiting these opportunities is not straightforward because both task selection and early stopping leave some final outcomes unobserved, even though these outcomes define the agent's benchmark performance. Simply skipping predictable tasks or terminating predictable trajectories can save computation, but the resulting observed outcomes are no longer the full set that a standard benchmark evaluation would use. A cost-efficient evaluator therefore has to decide where execution can

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.

be reduced while simultaneously accounting for the outcomes it chooses not to observe. This leads to three coupled questions: what tasks should we select to evaluate, when should we stop an initiated task once its partial trajectory becomes sufficiently informative, and how can we estimate performance accurately from the resulting incomplete observations?

We introduce **ACES**, an **A**gentic **C**ost-**E**fficient Evaluation framework with **S**tatistical guarantees, an evaluator agent that manages the entire evaluation process from allocating evaluation effort (in terms of which tasks to select and when to stop) to estimating final performance. *ACES works in a closed loop*. At each evaluation batch, it observes the current evaluation state, updates its outcome predictor and evaluation policy, decides which tasks to initiate, and monitors each initiated trajectory to determine whether further execution is worth its cost. It then uses the randomized evaluation decisions and observed trajectory information to estimate the target agent's overall performance. The newly collected observations return to the evaluation state and inform the next batch, so ACES progressively refines both its evaluation decisions and its performance estimate. We summarize our contribution as below:

- **Controllable agentic evaluation.** ACES is a novel agentic evaluation framework in which decisions about what to evaluate and when to stop are governed by a learned randomized policy under explicit evaluation-cost constraints.
- **Statistically valid performance evaluation.** We develop an estimator that corrects for adaptive task selection and early stopping, preserves unbiased performance estimation and supports valid uncertainty quantification.
- **Theoretical insights for when early stopping helps.** We characterize when trajectory-level stopping improves efficiency beyond task selection alone.
- **Substantial empirical cost savings.** ACES reduces the cost required to reach a fixed estimation accuracy by 36% on ALFWorld and 26.7% on ScienceWorld, with consistent evidence across three agent benchmarks.

2 Problem setup: Agentifying AI Evaluation

AI agent evaluation is increasingly both an agentic and resource-intensive process. Recent work has begun to treat evaluation itself as an automated system that interacts with target agents and their environments, rather than as a passive procedure that only scores final outputs (Zhuge et al., 2025; Shi et al., 2026a). At the same time, evaluating target AI agents can be computationally expensive, as a single benchmark task may require a long interactive rollout involving repeated model calls, tool use, and environment interactions (Liu et al., 2024; Zhou et al., 2024; Xie et al., 2024; Li et al., 2026).

To understand where these costs arise and whether and how evaluation can be made more efficient, it helps to first spell out how a standard benchmark evaluation works. A target AI agent is often evaluated on a collection of n tasks. We allow these tasks to be evaluated in batches so that information collected from earlier batches can inform how subsequent tasks are evaluated. For each task i, the agent produces a trajectory $H_{i,0}, \ldots, H_{i,T_i}$, where T_i is the terminal stage of the run and $H_{i,t}$ records the information revealed at stage t (such as agent actions, tool outputs, and environment feedback). We write $\mathcal{H}_{i,t}$ for the information available through stage t and $H_{i,0}$ for the task information before execution. Because each additional stage may require further model calls, tool use, and environment interactions, evaluation cost accumulates along the trajectory. At the terminal stage T_i, the run produces a final outcome $Y_i \in \{0, 1\}$. For example, Y_i may indicate whether a coding agent passes the tests for a SWE-bench task (Jimenez et al., 2024). We aim to accurately capture the target agent's overall performance quantified through $\mu = \mathbb{E}[Y_i]$.

This evaluation process creates an opportunity to build a cost-effective agentic evaluator that manages how the benchmark is executed: rather than running every task to completion by default, the evaluator can use information available before and during execution to decide how evaluation resources should be allocated. The key challenge is how the evaluator should make these decisions to reduce token cost without compromising the accuracy of the resulting estimate of μ. This perspective leads to three questions:

- **What tasks to evaluate?** Can the evaluator use historical evaluation data (when available) and observations accumulated from earlier evaluation runs $\mathcal{D}_{<b}$, to determine which task outcomes Y_i are already predictable and focus execution on more informative tasks?

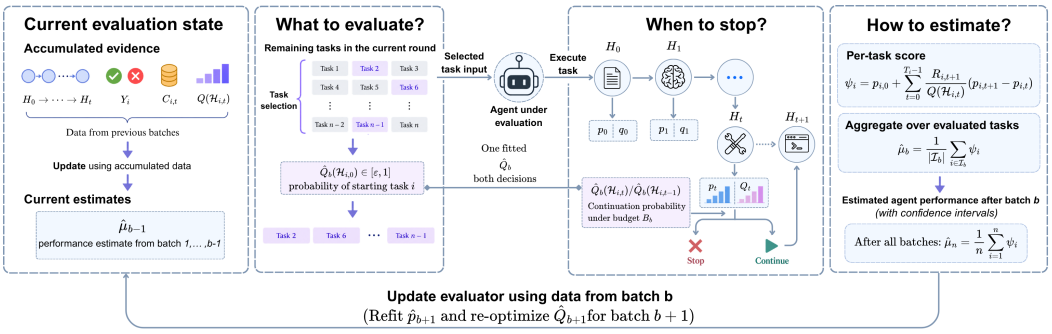


Figure 1: **Overview of ACES.** ACES adaptively decides what to evaluate and when to stop, while using bias-corrected task-level scores to estimate agent performance. Data collected in each batch update the predictor and continuation policy used in subsequent batches.

- **When to stop the task?** Can the evaluator use partial trajectory $\mathcal{H}_{i,t}$ to predict Y_i and stop before T_i when further execution is unlikely to provide enough information to justify its cost?
- **How to estimate μ accurately?** When some Y_i are unobserved because tasks are skipped or stopped early, how can the evaluator still estimate μ without bias while controlling variance?

Building on the view that the evaluation process itself can be instantiated as an AI agent, we develop **ACES** (**A**gentic **C**ost-**E**fficient Evaluation with **S**tatistical guarantees). ACES learns from historical data and information accumulated during evaluation to determine what to evaluate and when to stop, reducing unnecessary execution cost while preserving reliable estimation of μ in terms of unbiasedness, controlled variance, and valid uncertainty quantification. Figure 1 summarizes the resulting closed-loop evaluation workflow.

3 ACES: Agentic Cost-Efficient Evaluation with Statistical Guarantees

ACES uses predictions of final outcomes to allocate evaluation effort under the cost budget. A single cumulative policy governs both whether a task is started and how far its trajectory continues. The final performance estimator accounts for these decisions using the recorded observation probabilities.

Let $[n] = \{1, \ldots, n\}$ index all benchmark tasks, partitioned into batches $\mathcal{I}_1, \ldots, \mathcal{I}_W$ with expected-cost budgets $B_1, \ldots, B_W$ satisfying $\sum_{b=1}^{W} B_b = B$. We initialize learning for the target agent by running a randomly selected first batch $\mathcal{I}_1$ to completion using budget B_1. Before each subsequent batch b, ACES updates its predictor and policy using observations from earlier batches, denoted by $\mathcal{D}_{<b}$. Both are held fixed throughout the batch and applied to the initial task information and observed partial trajectories to determine which tasks to start and when to stop. Figure 1 summarizes this workflow, and Algorithm 1 in Appendix A gives the pseudocode for ACES.

3.1 What to Evaluate

ACES uses the initial task information and the corresponding outcome prediction to decide probabilistically whether each task is started. For task $i \in \mathcal{I}_b$, let $\widehat{p}_b$ denote the predictor fitted before batch b, and write

$$p_{i,t} = \widehat{p}_b(\mathcal{H}_{i,t}) \in [0,1], \qquad 0 \le t < T_i, \qquad p_{i,T_i} = Y_i.$$

The initial prediction $p_{i,0}$ uses only $H_{i,0}$ and is available for every task. At later stages, the predictor also uses the observed actions, tool outputs, and environment feedback. At completion, we replace the prediction with the observed outcome Y_i.

Let $Q^{(b)}(\mathcal{H}_{i,t})$ denote the cumulative probability of reaching stage $t+1$ along a given trajectory under the policy used in batch b. ACES starts task i with probability $Q^{(b)}(\mathcal{H}_{i,0})$. This starting probability and the continuation probabilities at later stages are learned together under the cost budget, as described in Section 3.2. The allocation reflects both prediction error and execution cost.

A task whose outcome is already predictable from its initial information may receive a lower starting probability when execution is costly.

We train the predictor using partial trajectories as inputs and their final outcomes as prediction targets. When historical complete trajectories are available, we first learn a shared representation $h_{\widehat{\phi}}$ and agent-specific prediction heads by minimizing squared prediction error. The shared representation captures common information in task descriptions and execution histories, while the prediction heads account for differences in agent performance. During target-agent evaluation, we hold the learned representation fixed. Before batch b, we update the target-agent prediction head using completed target-agent runs from all preceding batches, giving $\widehat{p}_b(\mathcal{H}) = p_{\widehat{\theta}_b}(h_{\widehat{\phi}}(\mathcal{H}))$.

Let $R_{i,t} \in \{0,1\}$ indicate whether stage t is observed, with $R_{i,0} = 1$ because the initial task information is always available. A completed run from batch r has observation probability $Q^{(r)}(\mathcal{H}_{i,T_i-1})$. We weight its training loss by the inverse of this probability to account for earlier task selection and stopping (Horvitz and Thompson, 1952).

For each completed run, the summed prediction loss is $g_i = \sum_{s=0}^{T_i-1}\{Y_i - p_\theta(h_{\widehat{\phi}}(\mathcal{H}_{i,s}))\}^2$. We update the prediction head by minimizing its weighted empirical average,

$$\widehat{\mathbb{E}}_{b-1}[g] = \frac{1}{N_{b-1}} \sum_{r=1}^{b-1} \sum_{\substack{i \in \mathcal{I}_r: \\ R_{i,T_i}=1}} \frac{g_i}{Q^{(r)}(\mathcal{H}_{i,T_i-1})}, \qquad N_{b-1} = \sum_{r=1}^{b-1} |\mathcal{I}_r|. \tag{1}$$

Here N_{b-1} counts all tasks assigned to earlier batches, including those not started or stopped early. The weighting gives more weight to trajectories that were less likely to be completed. We use the same weighted averaging operation below to evaluate prediction errors and execution costs for a proposed policy.

3.2 When to Stop

Early stopping is governed by the same cumulative policy that determines whether a task is started. At an observed nonterminal stage t, where $R_{i,t} = 1$, ACES decides whether to execute stage $t+1$. Here $Q^{(b)}(\mathcal{H}_{i,t})$ is the cumulative probability, from the beginning of the run, of reaching stage $t+1$. Conditional on having observed stage t, the probability of continuing is the ratio of this cumulative probability to that of reaching stage t. To make these decisions implementable sequentially, we set $\mathcal{H}_{i,-1} = \varnothing$ and $Q(\varnothing) = 1$, and require $\epsilon \le Q(\mathcal{H}_{i,t}) \le Q(\mathcal{H}_{i,t-1})$ for a fixed $\epsilon > 0$. Cumulative probabilities cannot increase because reaching a later stage requires observing all preceding stages. The lower bound preserves a positive probability of completing every run. Let $\mathcal{Q}_\epsilon$ denote the class of policies satisfying these requirements.

Before batch b, we apply the updated predictor to the partial trajectories of previously completed runs and compare its predictions with the observed final outcomes. These runs provide prediction errors at each stage and the corresponding execution costs. Let $C_{i,s+1} \ge 0$ denote the incremental cost of moving from stage s to stage $s+1$, such as the additional tokens consumed. For each completed trajectory, write $\widehat{p}_{b,s} = \widehat{p}_b(\mathcal{H}_s)$ for the updated prediction, suppressing the task index. The policy head Q_η uses the same fixed trajectory representation as the prediction head, with parameters specific to the target agent.

Reducing continuation probabilities saves execution cost but leaves more prediction error unresolved. When a partial trajectory already predicts the outcome well and substantial execution cost remains, stopping can free budget for other tasks. We evaluate this trade-off for a candidate policy Q_η using

$$\widehat{V}_{b-1}(\eta) = \widehat{\mathbb{E}}_{b-1}\left[\sum_{s=0}^{T-1}\left\{\frac{1}{Q_\eta(\mathcal{H}_s)} - \frac{1}{Q_\eta(\mathcal{H}_{s-1})}\right\}(Y - \widehat{p}_{b,s})^2\right],$$

$$\widehat{C}_{b-1}(\eta) = \widehat{\mathbb{E}}_{b-1}\left[\sum_{s=0}^{T-1} Q_\eta(\mathcal{H}_s) C_{s+1}\right].$$

In $\widehat{V}_{b-1}(\eta)$, each increase in inverse cumulative probability is weighted by that stage's squared prediction error. This estimates the additional variance from task selection and early stopping (Sec-

tion 3.3). In $\widehat{C}_{b-1}(\eta)$, each stage's cost is weighted by its probability of being incurred. Both criteria use the recorded completion probabilities in Equation (1) to account for earlier evaluation decisions.

We choose the policy that minimizes the empirical variance criterion within the batch's expected-cost budget,

$$\widehat{\eta}_b \in \operatorname*{argmin}_{\eta:\, Q_\eta \in \mathcal{Q}_\epsilon} \widehat{V}_{b-1}(\eta) \qquad \text{subject to} \qquad |\mathcal{I}_b|\, \widehat{C}_{b-1}(\eta) \le B_b, \tag{2}$$

and set $Q^{(b)} = Q_{\widehat{\eta}_b}$. The constraint is imposed on estimated expected cost. This single optimization determines both the starting probabilities and the continuation probabilities at later stages.

For a fixed predictor and a known distribution of trajectories, outcomes, and costs, the corresponding oracle policy solves the following problem, with the task index suppressed,

$$\begin{aligned} \underset{Q(\cdot)}{\text{minimize}} \quad & \mathbb{E}\left[\sum_{s=0}^{T-1}\left\{\frac{1}{Q(\mathcal{H}_s)} - \frac{1}{Q(\mathcal{H}_{s-1})}\right\}(Y - p_s)^2\right] \\ \text{subject to} \quad & n\,\mathbb{E}\left[\sum_{s=0}^{T-1} Q(\mathcal{H}_s) C_{s+1}\right] \le B, \\ & \epsilon \le Q(\mathcal{H}_s) \le Q(\mathcal{H}_{s-1}), \qquad 0 \le s < T. \end{aligned} \tag{3}$$

Equation (2) implements this objective batch by batch using the updated predictor and weighted estimates from earlier completed runs.

During each new run, ACES applies the fitted policy at each observed nonterminal stage t, continuing to stage $t+1$ with probability

$$\Pr\Big(R_{i,t+1} = 1 \mid R_{i,t} = 1, \mathcal{H}_{i,t}, Q^{(b)}\Big) = \frac{Q^{(b)}(\mathcal{H}_{i,t})}{Q^{(b)}(\mathcal{H}_{i,t-1})}. \tag{4}$$

At $t = 0$, the denominator is one, so this is the starting probability $Q^{(b)}(\mathcal{H}_{i,0})$. After each subsequent observed stage, ACES evaluates the same fitted policy on the updated partial trajectory and makes a fresh randomized continuation decision. If the decision is to stop, no further stages are executed. Otherwise, execution continues until the next decision or until the run reaches T_i, when Y_i is observed. The policy uses only the information observed so far in each run.

We record the observed trajectories, incurred costs, and actual sampling probabilities, together with the final outcomes of completed runs. These records support subsequent prediction and policy updates and the final performance estimate. A two-batch implementation applies the predictor and policy fitted after the first batch to all remaining tasks. Both batches contribute to the final estimator.

3.3 How to Estimate

For each task $i \in \mathcal{I}_b$, ACES constructs a task-level score and averages these scores over all n tasks to estimate performance,

$$\widehat{\mu}_n = \frac{1}{n}\sum_{b=1}^{W}\sum_{i\in\mathcal{I}_b}\psi_i^{(b)}, \qquad \psi_i^{(b)} = \underbrace{p_{i,0}}_{\text{initial prediction}} + \underbrace{\sum_{s=0}^{T_i-1}\frac{R_{i,s+1}}{Q^{(b)}(\mathcal{H}_{i,s})}(p_{i,s+1} - p_{i,s})}_{\text{sequential bias correction}}, \qquad i \in \mathcal{I}_b. \tag{5}$$

Every task contributes its initial prediction $p_{i,0}$. When stage $s+1$ is observed, so that $R_{i,s+1} = 1$, the score adds the prediction change $p_{i,s+1} - p_{i,s}$ divided by $Q^{(b)}(\mathcal{H}_{i,s})$, the probability that this transition is observed. The sum is computed only over observed transitions, so an unstarted task contributes only $p_{i,0}$. Each score uses the predictor fixed before its batch and the probabilities actually used to evaluate that task. We omit the batch superscript when writing the scores below.

Using the last prediction alone for a missing outcome can introduce bias when the predictor is imperfect. Since $p_{i,T_i} = Y_i$, the prediction changes telescope to give $p_{i,0} + \sum_{s=0}^{T_i-1}(p_{i,s+1} - p_{i,s}) = Y_i$. Inverse-probability weighting recovers these changes in expectation, giving $\mathbb{E}[\psi_i] = \mu$ even when the predictor is imperfect. In the fully observed pilot batch, all cumulative probabilities equal one and the score reduces to Y_i.

For a fixed predictor and policy, suppressing the task index, the score has variance

$$\mathrm{Var}(\psi) = \mathrm{Var}(Y) + \mathbb{E}\left[\sum_{s=0}^{T-1}\left\{\frac{1}{Q(\mathcal{H}_s)} - \frac{1}{Q(\mathcal{H}_{s-1})}\right\}(Y-p_s)^2\right]. \tag{6}$$

The outcome variance $\mathrm{Var}(Y)$ does not depend on the policy. The second term is the additional variance from task selection and early stopping. It is the objective in Equation (3), and its empirical counterpart is $\widehat{V}_{b-1}(\eta)$. Thus, the policy optimization in Section 3.2 directly connects the allocation of execution cost to the precision of the final performance estimate.

After all batches are completed, we estimate uncertainty from the task-level scores. The estimated variance and an asymptotic $(1-\alpha)$ confidence interval are

$$\widehat{\mathrm{Var}}(\widehat{\mu}_n) = \frac{1}{n(n-1)}\sum_{i=1}^{n}(\psi_i - \widehat{\mu}_n)^2, \qquad \widehat{\mu}_n \pm z_{1-\alpha/2}\sqrt{\widehat{\mathrm{Var}}(\widehat{\mu}_n)}, \tag{7}$$

where $z_{1-\alpha/2}$ is the corresponding standard normal quantile. Proposition 1 establishes unbiasedness under the stated sampling conditions and asymptotic validity of this interval when the fitted models stabilize.

4 Theoretical Investigation

In this section, we establish the theoretical properties of the methodology developed in Section 3. We begin with two assumptions.

Assumption 1 (Independent task runs)**.** If every task were run to completion by the target agent, the resulting trajectory, outcome, and execution costs, $(\mathcal{H}_{i,T_i}, Y_i, (C_{i,s})_{s=1}^{T_i})$, would be independent and identically distributed across tasks.

Assumption 2 (Stability of fitted models)**.** As earlier target-agent data accumulate, the fitted models converge on each nonterminal prefix of an independent run to deterministic limits

$$\widehat{p}_{b,s} \xrightarrow{p} p^*(\mathcal{H}_s), \qquad Q^{(b)}(\mathcal{H}_s) \xrightarrow{p} Q^*(\mathcal{H}_s). \tag{8}$$

Assumption 1 is natural when tasks are sampled from the same benchmark distribution and run in separate environments. Assumption 2 supports reliable uncertainty estimates and is plausible when larger training samples reduce sensitivity to individual runs. With V_* denoting the variance in Equation (6) evaluated at p^* and Q^*, we obtain the following estimation and inference guarantees.

Proposition 1 (Adaptive statistical inference)**.** *Under Assumption 1 and the sampling conditions in Appendix E.1, $\widehat{\mu}_n$ is unbiased and $\mathbb{E}[(\widehat{\mu}_n - \mu)^2] \le (n\epsilon^2)^{-1}$. If Assumption 2 and the batch condition in Equation* (12) *also hold, then unconditionally*

$$\sqrt{n}(\widehat{\mu}_n - \mu) \Rightarrow N(0, V_*), \qquad n\,\widehat{\mathrm{Var}}(\widehat{\mu}_n) \xrightarrow{p} V_*. \tag{9}$$

When $V_* > 0$, the confidence interval in Section 3.3 is asymptotically valid.

We next characterize when early stopping improves efficiency beyond task selection alone. For notational convenience, define the remaining prediction error at stage s as $r_s = \mathbb{E}[(Y-p_s)^2 \mid \mathcal{H}_s]$, and the expected remaining execution cost as $k_s = \mathbb{E}[\sum_{j=s+1}^{T} C_j \mid \mathcal{H}_s]$, for $s < T$. Let $\bar{B} = B/n$ denote the expected budget per task. Among policies that complete every started run, let $\widetilde{Q}^*(H_0)$ denote the optimal probability of starting a task under this budget. The corresponding optimization problem is given in Equation (17).

Proposition 2 (Value of early stopping)**.** *For a fixed predictor, under Assumption 1 and the interior allocation and regularity conditions stated in Appendix E.2, there exists an early-stopping policy in $\mathcal{Q}_\epsilon$ that has strictly smaller estimator variance than the optimal task-selection policy $\widetilde{Q}^*$ at the same expected cost if and only if some stage $s \ge 1$ satisfies*

$$\frac{r_s}{k_s} < \frac{r_0}{k_0} \quad \textit{with positive probability on } \{s < T\}. \tag{10}$$

If no such stage exists, $\widetilde{Q}^*$ is also optimal for Equation (3)*. Moreover, $\widetilde{Q}^*$ has strictly smaller estimator variance than uniform random sampling whenever r_0/k_0 varies across tasks.*

Intuitively, the condition says that early stopping is helpful when an observed partial trajectory already predicts the final outcome accurately, but completing the run would still require substantial execution cost. The stopping policy can assign a lower continuation probability to such runs and use the saved budget to start more tasks. This reallocation reduces the estimator variance at the same expected cost. If partial trajectories never provide sufficiently accurate predictions before most of the execution cost has been incurred, early stopping cannot improve upon task selection alone.

Remark 1 (Normalized exponential decay). Suppose runs have a common known length $T \geq 2$ and constant positive cost per stage, with $r_s/r_0 = e^{-\kappa s/T}$ before completion. Larger $\kappa > 0$ means that prediction error falls more steeply as execution progresses. Substituting this curve into Equation (10) shows that, under Assumption 3, early stopping improves efficiency if and only if

$$\kappa > -T \log(1 - 1/T). \tag{11}$$

The threshold is close to 1 when T is large. A sufficiently steep decline in prediction error therefore makes it worthwhile to stop some runs and use the saved budget to evaluate additional tasks. Appendix D visualizes this condition across different prediction-error decay rates.

5 Empirical studies on Benchmark Data

In this section, we study two sources of efficiency gains in agent evaluation: task selection and trajectory-level early stopping. Even without stopping, task-aware sampling improves over random full-run evaluation; early stopping further reduces error by using information revealed during execution. We demonstrate these gains on `ALFWorld` and `ScienceWorld`, with additional `SWE-bench` results in the Appendix. Together, task-aware sampling and early stopping reduce estimation error at fixed cost and execution cost at fixed precision.

5.1 Benchmark studies

ALFWorld. ALFWorld provides text-based household environments in which agents execute sequences of actions to satisfy natural-language goals (Shridhar et al., 2020), containing 2,130 recorded runs from 34 models, covering 355 task identifiers and six retrieval conditions. Each run has a binary outcome Y_i indicating task success. The dataset has a mean success rate of 0.455. To examine a setting with moderately informative initial covariates, we augment the recorded task-category variables with a generated covariate $Z_i = 0.5(2Y_i - 1) + \varepsilon_i, \varepsilon_i \overset{\text{iid}}{\sim} \mathcal{N}(0, 1)$. The predictor combines task-category indicators, Z_i, and behavioral features summarizing action count, progress relative to the configured action limit, pickup and placement behavior, special actions, repetition, action diversity, navigation, and observation length. We standardize features using the training partition and fit a gradient-boosting classifier. We then adopt an information schedule: $\widetilde{p}_{i,t} = \widehat{p}_{i,\min\{2t,T_i\}}, t = 0, \ldots, T_i$, where $\widehat{p}_{i,t}$ is the fitted prediction based on the original prefix.

We compare ACES with three benchmarks. *Full trajectories+random subset* uniformly samples runs without replacement, executes each to completion, and estimates the bank-level success rate from observed outcomes. *Without early stopping* uses the same learned initial sampling and prediction-based correction as ACES but completes every selected run; comparison with random sampling captures the benefit of selection and correction, while comparison with ACES isolates the gain from within-trajectory stopping. We also compare with Factorized Active Querying (FAQ) (Wu et al., 2026), which receives the same initial predictions as score factors and samples full trajectories with replacement, charging repeated queries their full execution cost. We report checkpoint-level success/failure classification accuracy and RMSE of the estimated evaluation-bank success rate, in percentage points. Execution cost is based on a character-level token proxy over recorded actions, observations, and reasoning, normalized by the cost of fully executing all evaluation runs. For cost efficiency, we invert the RMSE–cost curves and report the expected cost required to reach one percentage point RMSE.

Figure 2 demonstrates the complementary benefits of task allocation and trajectory-level early stopping. In panel (b), the "without early stopping" ablation achieves lower RMSE than random full-run sampling at all reported budgets, reflecting gains from task selection and prediction-based correction. ACES further reduces RMSE by leveraging partial-trajectory information to decide which runs to continue, as supported by the early predictive accuracy in panel (a). Figure 2 (c) translates

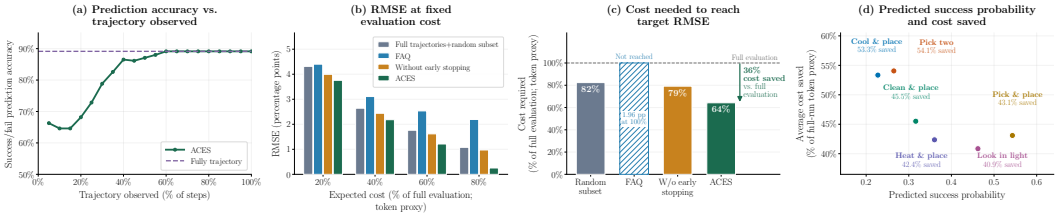


Figure 2: ALFWorld study results: (a) prediction accuracy versus observed trajectory, (b) RMSE vs expected cost, (c) cost reduction, and (d) predicted success probability and saved costs across six task categories.

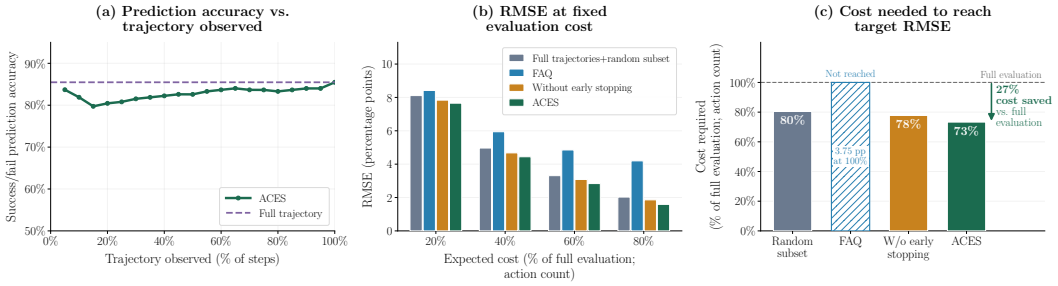


Figure 3: ScienceWorld study: (a) prediction accuracy (b) RMSE comparison (c) Cost needed to reach target RMSE.

these gains into computational savings. To achieve an RMSE of one percentage point, ACES requires only 64% of the full-evaluation cost, yielding a 36% reduction. FAQ does not reach this target within 100% expected cost and has an RMSE of approximately 1.96 percentage points at that budget, potentially due to its sampling-with-replacement scheme. Panel (d) further shows that the resulting cost savings vary across task categories. Overall, informative task covariates and partial trajectories provide complementary gains in evaluation efficiency.

ScienceWorld. ScienceWorld is an interactive text environment where agents complete scientific tasks through action–observation sequences (Wang et al., 2022). We use 387 recorded runs from seven agents, spanning 97 task variations, 65 task descriptions, and 30 task categories. Intermediate environment scores provide execution-time progress feedback. We define success as a final score of 100, yielding 71 successes and 316 failures (82% failure rate). Initial predictions use the task description, task category, and agent identity. Prefix predictions additionally use observed actions and observations, the current environment score, and behavioral summaries such as stagnation, repetition, and invalid-action frequency. We preserve the observed timing of all information and introduce no generated covariates, assuming intermediate scores are available to the evaluator during execution. Text features use TF-IDF followed by 64-dimensional SVD. We fit logistic regression for initial predictions and gradient-boosted trees for prefix predictions. ACES is compared with the same benchmark designs and evaluation metrics as in the `ALFWorld` study. Figure 3(a) shows that `ScienceWorld` has smaller, more gradual accuracy gains than `ALFWorld`, consistent with its more modest incremental RMSE improvements. In Figure 3(b), the "Without early-stopping" ablation outperforms random full-run sampling at all budgets, while ACES performs best; at 40% expected action cost, ACES achieves the lowest RMSE. Figure 3(c) shows that, at a target RMSE of two percentage points, ACES requires the least computation, using about 73% of full-evaluation action cost and saving 27%.

Across the `ALFWorld` and `ScienceWorld` studies, ACES achieves the lowest RMSE at the reported budgets and the largest cost savings among the compared designs. However, the incremental benefit of early stopping is more modest in `ScienceWorld`, suggesting that the contributions of initial information and early stopping depend on the setting. This variation motivates us to ask the question: Under what conditions do early stopping and initial information help the most?

5.2 When do early stopping and initial information help?

To investigate under what settings early stopping and initial information help, we conduct two experiments using the `ALFWorld` dataset: one changes when information becomes available during execution, and the other changes the information available before execution. We summarize the results in Figure 4.

Information timing. Figure 4 (a)-(b) isolate the timing of information while holding initial and full-trajectory predictions fixed. We construct $\widehat{p}_{i,t}^{(a)} = \widehat{p}_{i,\lfloor T_i(t/T_i)^{1/a} \rfloor}, a \in \{0.5, 1, 3\}$, where T_i is the recorded length of run i. The transformation delays information when $a < 1$, preserves its original timing when $a = 1$, and moves it earlier when $a > 1$. This is a controlled reassignment of recorded predictions to execution stages, and we do not change the outcomes or execution costs. Figure 4 (a) varies how quickly predictive information becomes available while holding the initial and endpoint predictions fixed. Panel (b) shows that earlier information yields larger RMSE reductions from early stopping, with the largest gain at $a = 3$. Thus, endpoint accuracy alone does not determine the value of stopping; useful information must arrive while substantial execution cost remains, consistent with Proposition 2. This also helps explain the benchmark results: `ALFWorld` provides informative predictions earlier, whereas `ScienceWorld` exhibits an initial accuracy decline followed by gradual recovery.

Initial information. Figure 4 (c)-(d) vary the strength of a generated pre-execution covariate: $Z_i^{(\delta)} = \delta(2Y_i - 1) + \varepsilon_i, \varepsilon_i \sim \mathcal{N}(0, 1), \delta \in \{0, 0.5, 1\}$. The information schedule is fixed throughout this sweep: the prediction originally available after $\min\{2t, T_i\}$ actions is supplied at stage t. Panel (c) shows that stronger initial signals improve prediction accuracy, as initial accuracy rises from $\delta = 0$ to $\delta = 1$. Panel (d) reports RMSE reductions relative to random full-trajectory sampling at matched expected cost. The RMSE reduction is the largest at the strongest signal strengths.

Together, the experiments distinguish two sources of efficiency: useful information before execution can improve prediction and allocation, while information acquired during execution can justify stopping only when its value arrives sufficiently early relative to its cost. This distinction explains why a design can improve upon random sampling even when its additional stopping gain is modest. It also provides additional insights on the benchmark studies: the relevant question is not simply how accurate a predictor becomes, but how much uncertainty it resolves before the remaining execution budget is spent.

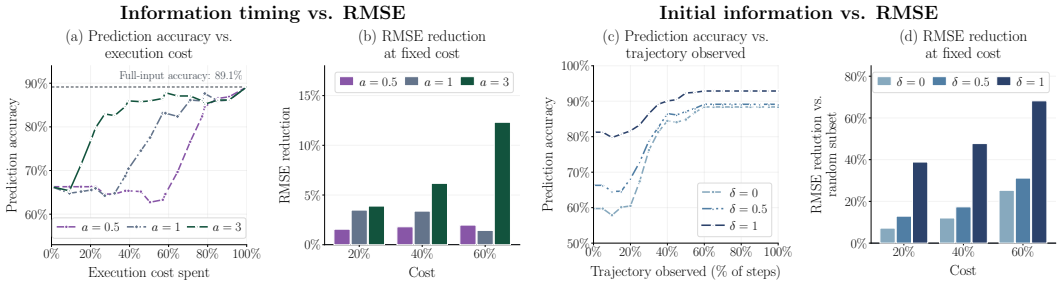


Figure 4: (a)–(b) Information timing: Larger a shifts predictive information earlier. (a) shows prediction accuracy versus execution cost incurred; (b) shows RMSE reduction relative to the "without early-stopping" ablation at matched expected cost. (c)–(d) Initial information: Larger δ strengthens initial information. (c) shows prediction accuracy along the trajectory; (d) shows RMSE reduction at matched expected cost.

AI USE STATEMENT

In this work, we used generative AI tools for none of the tasks with required disclosure. We have not used generative AI tools to generate synthetic data sets, help develop theoretical models or conceptual frameworks, formulate mathematical claims, provide critical ingredients for proving mathematical claims, assist in the writing of proofs, propose or refine hypotheses, design or provide feedback on research methodology or experiments, implement methods, clean or reformat datasets, or interpret results, and the rest of the required disclosure tasks are not applicable to this work. Additionally, we used generative AI tools to edit the research paper to improve readability, including grammar and wording. We have reviewed all AI-assisted work. All AI-assisted edits were reviewed by the authors to ensure that they did not alter the technical content or meaning of the paper. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REFERENCES

Angelopoulos, A. N., Bates, S., Fannjiang, C., Jordan, M. I., and Zrnic, T. (2023). Prediction-powered inference. *Science*, 382(6671):669–674.

Hall, P., Heyde, C. C., Birnbaum, Z., and Lukacs, E. (1980). *Martingale limit theory and its application*, volume 12. Elsevier.

Horvitz, D. G. and Thompson, D. J. (1952). A generalization of sampling without replacement from a finite universe. *Journal of the American statistical Association*, 47(260):663–685.

Jimenez, C. E., Yang, J., Wettig, A., Yao, S., Pei, K., Press, O., and Narasimhan, K. (2024). Swebench: Can language models resolve real-world github issues? In *International Conference on Learning Representations*, volume 2024, pages 54107–54157.

Kossen, J., Farquhar, S., Gal, Y., and Rainforth, T. (2021). Active testing: Sample-efficient model evaluation. In *International Conference on Machine Learning*, pages 5753–5763. PMLR.

Lanctot, M., Larson, K., Gemp, I., and Kaisers, M. (2026). Active evaluation of general agents: Problem definition and comparison of baseline algorithms. *arXiv preprint arXiv:2601.07651*.

Li, J., Zhao, W., Zhao, J., Zeng, W., Wu, H., Wang, X., Ge, R., Cao, Y., Huang, Y., Liu, W., et al. (2026). The tool decathlon: Benchmarking language agents for diverse, realistic, and long-horizon task execution. In *International Conference on Learning Representations*, volume 2026, pages 53359–53394.

Li, Y., Ma, J., Ballesteros, M., Benajiba, Y., and Horwood, G. (2024). Active evaluation acquisition for efficient llm benchmarking. *arXiv preprint arXiv:2410.05952*.

Lin, Y., Wang, Z., Liu, M., Shan, Y., Bai, L., Zhang, J., Jin, X., Chen, B., Su, J., Wang, X., et al. (2026). Bagen: Are llm agents budget-aware? *arXiv preprint arXiv:2606.00198*.

Liu, X., Yu, H., Zhang, H., Xu, Y., Lei, X., Lai, H., Gu, Y., Ding, H., Men, K., Yang, K., et al. (2024). Agentbench: Evaluating llms as agents. In *International Conference on Learning Representations*, volume 2024, pages 52989–53046.

Polo, F. M., Weber, L., Choshen, L., Sun, Y., Xu, G., and Yurochkin, M. (2024). tinybenchmarks: evaluating llms with fewer examples. *arXiv preprint arXiv:2402.14992*.

Ruan, K., Huang, Z., Zhou, Z., Wei, Q., Lin, J., Wang, X., and Sun, H. (2026). Doomed from the start: Early abort of llm agent episodes via a recall-controlled probe cascade. *arXiv preprint arXiv:2607.06503*.

Saranathan, G., Xu, C., Alam, M. P., Kumar, T., Foltin, M., Wong, S. Y., and Bhattacharya, S. (2025). Sublime: Subset selection via rank correlation prediction for data-efficient llm evaluation. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 30572–30593.

Shi, W., Wang, Y., Zhao, Y., Chen, Y., Feng, F., Hao, X., Su, X., Gu, Q., Su, H., Cai, X., et al. (2026a). Aj-bench: Benchmarking agent-as-a-judge for environment-aware evaluation. In *Findings of the Association for Computational Linguistics: ACL 2026*, pages 25371–25413.

Shi, Y., Sun, Z., Dong, J., Wan, C., Lo, D., and Gu, X. (2026b). Earlyeval: Cheaper agent evaluation via early outcome prediction. *arXiv preprint arXiv:2609.02783*.

Shridhar, M., Yuan, X., Côté, M.-A., Bisk, Y., Trischler, A., and Hausknecht, M. (2020). Alfworld: Aligning text and embodied environments for interactive learning. *arXiv preprint arXiv:2010.03768*.

Wang, R., Jansen, P., Côté, M.-A., and Ammanabrolu, P. (2022). Scienceworld: Is your agent smarter than a 5th grader? In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 11279–11298.

Wu, S., Nair, Y., and Candès, E. J. (2026). Efficient evaluation of llm performance with statistical guarantees. *arXiv preprint arXiv:2601.20251*.

Xie, T., Zhang, D., Chen, J., Li, X., Zhao, S., Cao, R., Hua, T. J., Cheng, Z., Shin, D., Lei, F., et al. (2024). Osworld: Benchmarking multimodal agents for open-ended tasks in real computer environments. *Advances in Neural Information Processing Systems*, 37:52040–52094.

Yang, J., Jimenez, C., Wettig, A., Lieret, K., Yao, S., Narasimhan, K., and Press, O. (2024). Swe-agent: Agent-computer interfaces enable automated software engineering. *Advances in Neural Information Processing Systems*, 37:50528–50652.

Zhou, S., Xu, F. F., Zhu, H., Zhou, X., Lo, R., Sridhar, A., Cheng, X., Ou, T., Bisk, Y., Fried, D., et al. (2024). Webarena: A realistic web environment for building autonomous agents. In *International Conference on Learning Representations*, volume 2024, pages 15585–15606.

Zhuge, M., Zhao, C., Ashley, D. R., Wang, W., Khizbullin, D., Xiong, Y., Liu, Z., Chang, E., Krishnamoorthi, R., Tian, Y., et al. (2025). Agent-as-a-judge: Evaluating agents with agents.

Zrnic, T. and Candès, E. J. (2024). Active statistical inference. *arXiv preprint arXiv:2403.03208*.

A ACES PROCEDURE

Algorithm 1 ACES

Input: Benchmark tasks $\{H_{i,0}\}_{i=1}^{n}$, budget B, and probability floor ϵ.

Partition $[n]$ into $\mathcal{I}_1, \ldots, \mathcal{I}_W$, choosing $\mathcal{I}_1$ randomly, with budgets $B_1, \ldots, B_W$ satisfying $\sum_{b=1}^{W} B_b = B$.
Fit $h_{\widehat{\phi}}$ and agent-specific prediction heads using historical complete trajectories when available.

Run $\mathcal{I}_1$ to completion and record trajectories, costs, Y_i, and cumulative probabilities $Q^{(1)} \equiv 1$.
for $b = 2, \ldots, W$:
  Update $\widehat{p}_b$ using completed target-agent runs in $\mathcal{D}_{<b}$ and Equation (1), holding $h_{\widehat{\phi}}$ fixed.
  Fit $Q^{(b)} = Q_{\widehat{\eta}_b}$ by Equation (2) under budget B_b.
  Hold $\widehat{p}_b$ and $Q^{(b)}$ fixed throughout the batch.
  for each $i \in \mathcal{I}_b$:
    Observe $H_{i,0}$ and set $R_{i,0} = 1$ and $p_{i,0} = \widehat{p}_b(\mathcal{H}_{i,0})$.
    Set $R_{i,1} = 1$ with probability $Q^{(b)}(\mathcal{H}_{i,0})$, and 0 otherwise.
    if $R_{i,1} = 1$:
      Execute stage 1 and set $t \leftarrow 1$.
      while stage t is nonterminal:
        Observe $\mathcal{H}_{i,t}$ and set $p_{i,t} = \widehat{p}_b(\mathcal{H}_{i,t})$.
        Set $R_{i,t+1} = 1$ with probability $\frac{Q^{(b)}(\mathcal{H}_{i,t})}{Q^{(b)}(\mathcal{H}_{i,t-1})}$, and 0 otherwise.
        if $R_{i,t+1} = 0$: stop the run and **break**.
        Execute stage $t+1$ and set $t \leftarrow t+1$.
      if the run completes: observe Y_i and set $p_{i,T_i} = Y_i$.
    Record observed trajectories, predictions, costs, actual cumulative probabilities, and Y_i when observed.
Compute $\psi_i^{(b)}$ and $\widehat{\mu}_n$ by Equation (5) using the recorded probabilities, with $\psi_i^{(1)} = Y_i$.
Compute $\widehat{\mathrm{Var}}(\widehat{\mu}_n)$ and the confidence interval by Equation (7).

B CONNECTION TO EXISTING LITERATURE

B.1 COMPARISON TO EXISTING LITERATURE

Efficient evaluation through task selection. A substantial literature reduces evaluation cost by selecting a subset of benchmark examples rather than evaluating the full test set. Active Testing formalizes this problem as selective acquisition of expensive test labels and uses importance weighting to correct the bias introduced by nonuniform sampling (Kossen et al., 2021). More recent work adapts this idea to large language models. TINYBENCHMARKS uses item-response models and carefully selected subsets to recover full-benchmark performance from a small number of examples (Polo et al., 2024). Factorized Active Querying (FAQ) further uses historical model–question outcomes to adaptively select benchmark questions and constructs confidence intervals with frequentist coverage guarantees (Wu et al., 2026). Active evaluation has also been studied for general agents, where the evaluator sequentially chooses which task–agent pair to sample to improve model ranking (Lanctot et al., 2026). These methods establish that evaluation effort need not be distributed uniformly across benchmark tasks. ACES builds on the same basic opportunity but considers a second allocation decision that arises specifically for long-horizon agents: after a task has been selected, the evaluator can also decide how much of its trajectory should be executed.

Statistical inference under adaptive evaluation. Our statistical formulation is related to work on prediction-assisted (Angelopoulos et al., 2023) and active statistical inference, but the sequential structure of agent trajectories creates a different observation problem. Active Statistical Inference uses model predictions to prioritize which outcomes to acquire while retaining valid statistical inference under adaptive data collection (Zrnic and Candès, 2024). FAQ develops this perspective for finite-population LLM benchmarking and adaptively selects complete question outcomes while preserving confidence-interval coverage (Wu et al., 2026). In these settings, the central decision is whether a final outcome should be acquired. In ACES, a task instead generates a sequence of increasingly informative observations, and the evaluator may stop at any point along this sequence. The resulting estimator therefore corrects not only for whether a task is selected but also for the

probability that each successive part of its trajectory is observed. This sequential correction links the learned continuation policy directly to the variance of the final performance estimator.

Early stopping within agent trajectories. A complementary literature reduces cost within individual agent runs by predicting outcomes before execution is complete. EarlyEval predicts eventual success or failure from intermediate agent behavior and stops trajectories once a calibrated classifier becomes sufficiently confident; across SWE-bench Verified, TerminalBench, and Toolathlon, it reports substantial reductions in steps and token usage with small empirical changes in aggregate resolve rates (Shi et al., 2026b). Related work shows that eventual failures can be detected even earlier from internal model representations and develops recall-controlled probe cascades that terminate likely failed episodes (Ruan et al., 2026). Other recent work studies whether agents can recognize when a task is unlikely to finish within a remaining resource budget and uses this signal for early termination (Lin et al., 2026). These methods demonstrate that partial trajectories can contain actionable information well before completion. Their primary objective, however, is to identify trajectories that can be terminated while controlling prediction error, false-abort risk, or task-level performance degradation. ACES instead treats stopping as part of a benchmark-level statistical design: continuation probabilities are chosen according to their effect on both execution cost and the precision of the final performance estimate, and randomized stopping permits unbiased estimation even when the outcome predictor is imperfect.

Agentic approaches to evaluation. A separate line of work argues that the evaluator itself should become an agent rather than remain a passive scoring rule. Agent-as-a-Judge equips an evaluator with agentic capabilities so that it can assess intermediate steps and collect evidence about another agent's task-solving process (Zhuge et al., 2025). AJ-Bench further studies environment-aware judge agents that actively use tools and interact with environments to acquire evidence for verification (Shi et al., 2026a). These works expand what an automated evaluator can observe and verify. ACES focuses on a complementary form of agency: the evaluator controls the evaluation process itself. It decides which tasks receive computation, monitors initiated trajectories, decides when additional execution is worthwhile, estimates final performance from the selectively observed data, and uses newly acquired information to update subsequent evaluation decisions. The agentic component in ACES is therefore a closed-loop resource-allocation and estimation process rather than an automated replacement for a human or rule-based judge.

Relation of ACES to the closest existing approaches. ACES connects task-level adaptive evaluation and trajectory-level early stopping within a single statistically controlled evaluation loop. Relative to task-selection approaches such as FAQ, ACES introduces within-trajectory decisions and can save computation after a task has already been initiated. Relative to early-stopping approaches such as EarlyEval (Shi et al., 2026b), ACES does not require stopped trajectories to be treated as if their predicted outcomes were observed; randomized continuation and sequential correction instead account explicitly for the missing terminal outcomes. Relative to Agent-as-a-Judge approaches, ACES uses agency to control where evaluation effort is spent rather than primarily to improve the quality of the judging function. These distinctions lead to the central design principle of ACES: *what to evaluate, when to stop, and how to estimate cannot be optimized independently*. A learned randomized policy controls the first two decisions under an explicit evaluation budget, while the corresponding estimator uses the same sampling probabilities to preserve valid performance estimation.

C Additional empirical results

C.1 SWE-bench study

In this study, we evaluate the proposed design on recorded SWE-bench trajectories and summarize the takeaways as follows. First, short trajectory prefixes already provide useful outcome predictions, with additional observations yielding modest improvements in prediction accuracy. Second, the proposed design substantially improves estimation efficiency relative to the benchmark designs, while the no-stopping ablation retains most of this improvement. Therefore, the additional contribution of early stopping is modest and depends on the evaluation budget. Third, empirical coverage is close to nominal at moderate budgets, with more conservative intervals for the proposed design and its ablation at the highest reported budget.

Dataset. We use recorded agent trajectories on SWE-bench issues (Jimenez et al., 2024) from the public `nebius/SWE-agent-trajectories` dataset. The binary `target` field indicates whether the generated patch resolves the issue according to its evaluation tests. We use the first Parquet shard, containing 6,670 runs, and exclude trajectories with fewer than three parsed steps. Each step consists of an agent action and its following environment observation; the initial issue description forms the zero-step prefix. The target agent is `swe-agent-llama-70b`. Its 5,993 retained runs are shuffled using seed zero and divided into 2,996 predictor-training runs, 1,498 policy-development runs, and 1,499 evaluation runs. An additional 651 historical runs from the `swe-agent-llama-8b` and `swe-agent-llama-405b` agents support predictor training. The evaluation bank contains 272 successful runs, giving a mean success rate of 0.1815.

Study setup. For each prefix, we construct cumulative bag-of-words counts from the issue description and the actions and observations available so far. The vocabulary contains at most 250 terms. Counts undergo a $\log(1+x)$ transformation and standardization fitted on the training data. A shared multilayer perceptron with one 32-unit ReLU hidden layer and agent-specific logistic heads is trained by binary cross-entropy on historical and target-agent training runs. Shared-representation training uses at most 45,000 nonterminal prefix rows, 1,500 Adam updates, learning rate 0.01, and ℓ_2 coefficient 10^{-4}. We then freeze the representation and fit the target-agent logistic head for 1,200 updates, with learning rate 0.05 and ℓ_2 coefficient 10^{-3}. Predictions $\widehat{p}_{i,t}$ are clipped to $[0.001, 0.999]$ and estimate eventual success from the observed prefix. The execution policy uses a nested logistic head with separate initial selection and subsequent stopping coefficients. Define $u(p) = 4p(1-p)$ and $c(p) = 2|p - 1/2|$. Its features are $z_{i,0} = [1, u(\widehat{p}_{i,0}), c(\widehat{p}_{i,0})]^\mathsf{T}$, $d_{i,t} = \begin{bmatrix} u(\widehat{p}_{i,t}) - u(\widehat{p}_{i,0}) \\ c(\widehat{p}_{i,t}) - c(\widehat{p}_{i,0}) \\ t/60 \end{bmatrix}$. Writing σ for the logistic function, we set $\widetilde{Q}_{i,t+1} = \sigma(\alpha^\mathsf{T} z_{i,0} + \beta^\mathsf{T} d_{i,t})$, $Q_{i,0} = 1, Q_{i,t+1} = \min\{Q_{i,t}, \max(0.05, \widetilde{Q}_{i,t+1})\}$. Here $Q_{i,t}$ is cumulative survival through t steps, and the conditional next-step continuation probability is $q_{i,t} = Q_{i,t+1}/Q_{i,t}$. The fixed stage scale 60 does not require knowing a run's eventual length during execution. Both policy variants are fitted on all development runs by minimizing the exact conditional-variance objective subject to an expected-cost constraint. We use multistart sequential least-squares programming with analytic gradients and budgets from 10% to 95% in increments of 5 percentage points.

Methods in comparison. We compare the proposed design with three alternatives. "Random subsets" sample runs uniformly without replacement, execute each selected run fully, and estimate success rate using their mean outcome. "Without early stopping" sets $\beta = 0$ and fits the remaining initial-selection coefficients under the same development budget constraints. It shares the predictor, initial-selection features, and prediction correction with the full design, but every selected run is completed. *Factorized Active Querying (FAQ)* learns model and task factors and uses adaptive full-trajectory queries with prediction-assisted inverse-probability correction (Wu et al., 2026). Its factor model uses historical outcomes and the target-agent predictor-training split, without evaluation outcomes. Sampling governors are selected on the development split using 16 replays at each of two query budgets. The selected configuration is $(\beta_0, \rho, \gamma, \tau) = (1, 0, 0, 0.05)$, and evaluation uses 256 replays per query budget. Queries are drawn with replacement, and every draw, including a repeated query, incurs its full trajectory cost.

Evaluation metrics. At observed fraction x, prediction accuracy is evaluated after $k_i(x) = \mathrm{round}(xT_i)$ steps, classifying success when $\widehat{p}_{i,k_i(x)} > 0.5$ and weighting evaluation runs equally. The full-input reference uses the learned $\widehat{p}_{i,T_i}$. For the corrected estimator, the terminal value is instead the observed outcome Y_i; this substitution is not used to score prediction accuracy. Execution cost is the sum of action and observation character counts divided by four, expressed relative to the full evaluation-bank cost. This is a token-cost proxy and excludes predictor training, pilot acquisition, and prediction-compute overhead. The precision axis reports the nominal 95% half-width proxy $1.96 \times \mathrm{RMSE}$ in percentage points. Empirical interval coverage is evaluated separately. For coverage, we perform 10,000 execution-randomization replays per method and budget, holding the evaluation bank and fitted models fixed.

Study results. Figure 5 (a) shows that short prefixes already support useful predictions. This leaves relatively limited scope for additional predictive improvement from continued execution. Figure 5(b) shows substantial estimation gains over random subsets and adapted FAQ. However, the RMSE reduction of our proposed design relative to the "Without early stopping" ablation is rather marginal.

Table 1: SWE-bench: empirical coverage (%) of nominal 95% intervals. Parentheses report Monte Carlo standard errors.

Cost	Random	Without early stopping	Proposed	FAQ (adapted)
20%	95.79 (0.20)	95.82 (0.20)	95.75 (0.20)	95.04 (0.22)
40%	95.33 (0.21)	95.58 (0.21)	95.57 (0.21)	94.60 (0.23)
60%	95.25 (0.21)	95.86 (0.20)	95.76 (0.20)	94.67 (0.22)
80%	95.59 (0.21)	97.78 (0.15)	96.60 (0.18)	94.49 (0.23)

These comparisons indicate that initial selection together with prediction correction accounts for most of the observed improvement. FAQ improves over random subsets at lower budgets, while random sampling benefits increasingly from approaching a full census. Figure 5 (c) demonstrates the precision-cost trade-off, suggesting a similar pattern as panel (b). Panel (d) provides a descriptive diagnostic for the policy fitted at a 40% development budget. For each run, it pairs the prediction at the first binding stage with the minimum cumulative survival (or continuation probability), which equals Q_{i,T_i}, and averages these pairs within prediction bins. The continuation probability to the next stage is highest for intermediate predicted success probabilities and lower for confident outcomes, particularly confident failures. This is consistent with retaining uncertain runs. Table 1 suggests that our proposed design reaches nominal level coverage probability.

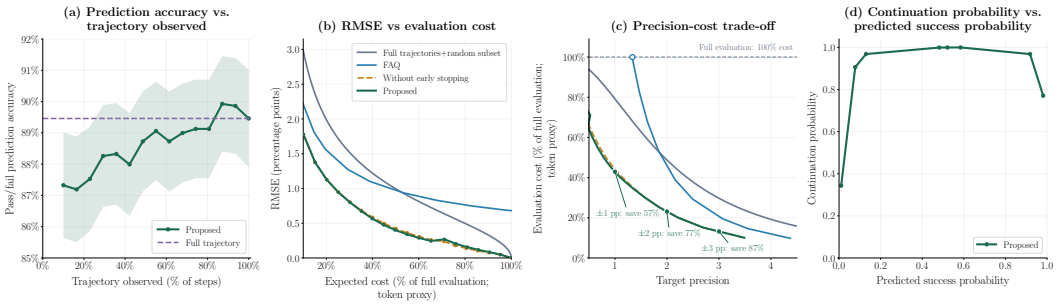


Figure 5: (a) Comparison of prediction accuracy with full trajectory prediction. (b) Comparison of RMSE. (c) Precision-cost trade-off. (d) Continuation probability versus predicted success probability.

C.2 Additional Section 5 study results

In this section, we provide the additional empirical results to complement the empirical study in Section 5. Specifically, we provide the coverage probabilities for ALFWorld study (Table 2) and ScienceWorld study (Table 3). The results suggest that the proposed estimator under our design attains nominal-level coverage.

Table 2: ALFWorld empirical coverage (%) probabilities. Parentheses report Monte Carlo standard errors.

Cost	Random	Without early stopping	Proposed	FAQ (adapted)
20%	94.75 (0.22)	95.16 (0.21)	95.46 (0.21)	94.90 (0.22)
40%	95.06 (0.22)	95.12 (0.22)	95.20 (0.21)	94.90 (0.22)
60%	95.41 (0.21)	95.01 (0.22)	94.80 (0.22)	94.91 (0.22)
80%	95.07 (0.22)	94.88 (0.22)	94.43 (0.26)	95.04 (0.22)

Table 3: ScienceWorld: empirical coverage (%) probabilities. Parentheses report Monte Carlo standard errors.

Cost	Random	Without early stopping	Proposed	FAQ (adapted)
20%	95.15 (0.12)	94.43 (0.13)	94.77 (0.13)	90.34 (0.17)
40%	95.03 (0.13)	94.57 (0.13)	95.22 (0.12)	93.01 (0.15)
60%	95.06 (0.13)	94.86 (0.13)	94.95 (0.13)	93.81 (0.14)
80%	95.13 (0.12)	94.72 (0.13)	94.56 (0.13)	93.94 (0.14)

D Illustration of Prediction Error Decay and Early Stopping Gain

We illustrate Remark 1 to show how rapidly prediction error must decline relative to execution cost for early stopping to improve efficiency. We consider a run with $T = 20$ equally costly stages. After stage s, the fraction of the full-run cost that remains is $k_s/k_0 = 1 - s/T$. We model the fraction of the initial prediction error that remains as $r_s/r_0 = e^{-\kappa s/T}$, where a larger κ represents a faster decline in prediction error. Early stopping can improve efficiency when the prediction error falls below the remaining execution cost at some nonterminal stage. Remark 1 gives the critical decay rate $\kappa_{\text{crit}} = -T\log(1 - 1/T) \approx 1.026$. Figure 6 shows decay rates below, at, moderately above, and well above this threshold. At or below the threshold, no nonterminal stage satisfies the strict gain condition. Above the threshold, the improvement region expands as prediction error declines more rapidly.

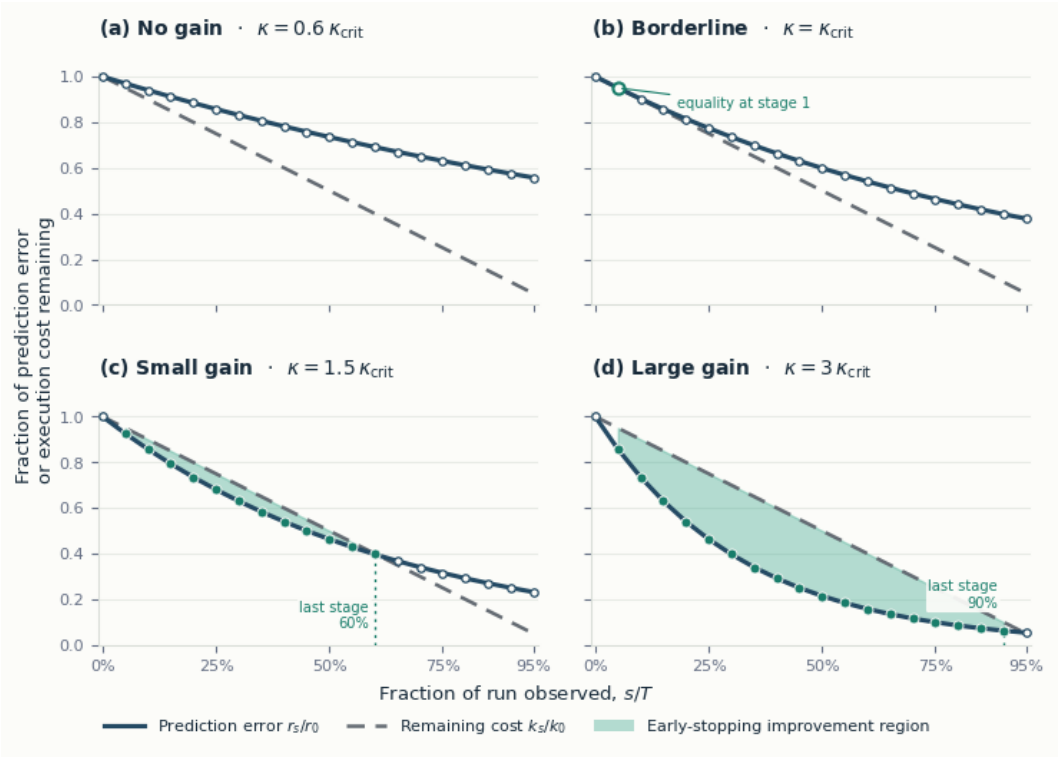


Figure 6: Normalized prediction error and remaining execution cost under exponential decay. The shaded regions and green points mark nonterminal stages satisfying $r_s/r_0 < k_s/k_0$, equivalently the condition in Equation (10). The dotted line marks the final stage satisfying this condition. The four panels use $\kappa/\kappa_{\text{crit}} \in \{0.6, 1, 1.1, 3\}$. The shaded area shows the potential for variance reduction.

E PROOFS OF THE MAIN RESULTS

E.1 PROOF OF PROPOSITION 1

Proof. We first specify the sampling and convergence conditions underlying Assumptions 1–2. Complete runs have $T_i < \infty$ almost surely and are independent of the historical data and the batch partition. Batch sizes are fixed in advance, and all assigned tasks, including unstarted tasks, are counted in n. Write $N_{b-1} = \sum_{r<b} |\mathcal{I}_r|$. Convergence in Equation (8) is unconditional over the historical data, earlier batches, fitting randomness, and an independent complete run, along every sequence of deployed batches with $N_{b-1} \to \infty$. The condition that early fits affect a vanishing fraction of tasks means

$$\frac{1}{n}\sum_{b=1}^{W} |\mathcal{I}_b|\, \mathbf{1}\{N_{b-1} < M\} \longrightarrow 0 \qquad \text{for every fixed } M < \infty. \tag{12}$$

Budgets and batch sizes may vary with n, while ϵ remains fixed. For two batches, this condition is met when the pilot size grows but its fraction of the total sample tends to zero.

Order tasks by batch, and let $\mathcal{F}_{i-1}$ contain the historical data, records from tasks preceding i, and fitting randomness available before task i. For $i \in \mathcal{I}_b$, the fitted functions are determined by $\mathcal{F}_{N_{b-1}}$. Let Z_i denote the complete run that would be generated without early stopping, including its trajectory, costs, and outcome. It is independent of $\mathcal{F}_{i-1}$. Conditional on $(\mathcal{F}_{i-1}, Z_i)$, the predictor and policy values along this run are fixed. Suppressing i, define $\rho_s = R_s/Q^{(b)}(\mathcal{H}_{s-1})$, with $\rho_0 = 1$. Fresh continuation randomization makes these weights a martingale with conditional mean one. Therefore

$$\mathbb{E}[\psi_i \mid \mathcal{F}_{i-1}, Z_i] = p_{i,0} + \sum_{s<T_i} (p_{i,s+1} - p_{i,s}) = Y_i. \tag{13}$$

Let S_i be the last observed stage. Summation by parts in the score gives

$$\psi_i = \frac{p_{i,S_i}}{Q^{(b)}(\mathcal{H}_{i,S_i-1})} - \sum_{s=0}^{S_i-1} p_{i,s}\left\{\frac{1}{Q^{(b)}(\mathcal{H}_{i,s})} - \frac{1}{Q^{(b)}(\mathcal{H}_{i,s-1})}\right\}.$$

Using $p_{i,s} \in [0,1]$ and the cumulative probability floor yields

$$1 - \epsilon^{-1} \le \psi_i \le \epsilon^{-1}, \qquad |\psi_i - \mu| \le \epsilon^{-1}. \tag{14}$$

Thus Equation (13) can be integrated to obtain $\mathbb{E}[\psi_i \mid \mathcal{F}_{i-1}] = \mu$. The centered scores $\xi_i = \psi_i - \mu$ are bounded martingale differences, proving unbiasedness and

$$\mathbb{E}[(\widehat{\mu}_n - \mu)^2] = \frac{1}{n^2}\sum_{i=1}^{n} \mathbb{E}[\xi_i^2] \le \frac{1}{n\epsilon^2}.$$

For the normal limit, summation by parts along the complete trajectory gives $\psi - Y = \sum_{s<T}(\rho_{s+1} - \rho_s)(Y - p_s)$. Conditional on the earlier history and the complete run, these martingale increments are orthogonal, with second moments $1/Q^{(b)}(\mathcal{H}_s) - 1/Q^{(b)}(\mathcal{H}_{s-1})$. The law of total variance therefore gives

$$\begin{aligned} v_i &:= \mathbb{E}[\xi_i^2 \mid \mathcal{F}_{i-1}] \\ &= \mathrm{Var}(Y) + \mathbb{E}\left[\sum_{s<T_i}\left\{\frac{1}{Q^{(b)}(\mathcal{H}_{i,s})} - \frac{1}{Q^{(b)}(\mathcal{H}_{i,s-1})}\right\}(Y_i - p_{i,s})^2 \,\middle|\, \mathcal{F}_{i-1}\right]. \end{aligned} \tag{15}$$

The sum inside this expectation lies between zero and $\epsilon^{-1} - 1$. For any fixed K, convergence of the fitted functions on the finitely many prefixes with $s < K$ implies convergence in probability of the sum on $\{T \le K\}$. Letting $K \to \infty$ and using $T < \infty$ almost surely gives convergence on a complete run. The uniform bound makes this convergence hold in L^1 as well. Consequently, $\mathbb{E}[|v_i - V_*|] \to 0$ along every sequence of batches with $N_{b-1} \to \infty$.

Split the average conditional-variance deviations into batches with $N_{b-1} < M$ and those with $N_{b-1} \geq M$. The first contribution vanishes by Equation (12). The second becomes arbitrarily small as $M \to \infty$, by the convergence along every sequence just established. Hence

$$\frac{1}{n}\sum_{i=1}^{n} v_i \xrightarrow{p} V_*. \tag{16}$$

The bound $|\xi_i| \leq \epsilon^{-1}$ also gives the conditional Lindeberg condition, since for every $\delta > 0$ and all sufficiently large n,

$$\frac{1}{n}\sum_{i=1}^{n} \mathbb{E}[\xi_i^2 \mathbf{1}\{|\xi_i| > \delta\sqrt{n}\} \mid \mathcal{F}_{i-1}] = 0.$$

The martingale central limit theorem proves the claimed normal limit (Hall et al., 1980).

Finally, $\xi_i^2 - v_i$ are bounded martingale differences, so their average converges to zero in L^2. The variance estimator satisfies

$$n\,\widehat{\mathrm{Var}}(\widehat{\mu}_n) = \frac{n}{n-1}\left\{\frac{1}{n}\sum_{i=1}^{n}\xi_i^2 - (\widehat{\mu}_n - \mu)^2\right\} \xrightarrow{p} V_*.$$

Slutsky's theorem establishes the unconditional confidence-interval coverage. □

E.2 Proof of Proposition 2

The comparison uses the following condition on the optimal policy without early stopping.

Assumption 3 (Interior reference allocation)**.** The optimal initial allocation uses the available budget and stays away from the probability bounds. For some $\delta > 0$,

$$\mathbb{E}[\widetilde{Q}^*(H_0)k_0] = \bar{B}, \qquad \epsilon + \delta \leq \widetilde{Q}^*(H_0) \leq 1 - \delta \quad \text{almost surely.}$$

This condition leaves room to use savings from early stopping to start more tasks. The proof uses this flexibility to compare policies at the same expected cost.

Proof. Fix the predictor and target-run distribution. The comparison is over $\mathcal{Q}_\epsilon$. The regularity conditions for the ratio comparison are $r_0 > 0$ almost surely, $0 < k_s < \infty$ on $\{s < T\}$, and $\mathbb{E}[\sum_{j=1}^{T} C_j] < \infty$, with finite runs as in the preceding proof. These conditions make the error-to-cost ratios and expected budgets well defined.

The optimal policy without early stopping. A policy that starts task H_0 with probability $\widetilde{Q}(H_0)$ and completes every started run has $Q(\mathcal{H}_s) = \widetilde{Q}(H_0)$ for all $0 \leq s < T$. Equations (6) and (3) reduce its allocation problem to

$$\underset{\epsilon \leq \widetilde{Q}(H_0) \leq 1}{\text{minimize}} \quad \mathbb{E}[(\widetilde{Q}(H_0)^{-1} - 1)r_0] \qquad \text{subject to} \qquad \mathbb{E}[\widetilde{Q}(H_0)k_0] \leq \bar{B}. \tag{17}$$

For every bounded function $g(H_0)$ with $\mathbb{E}[g(H_0)k_0] = 0$, the perturbation $\widetilde{Q}^* + tg$ remains feasible for sufficiently small $|t|$ by Assumption 3. Optimality therefore gives $\mathbb{E}[r_0 g/(\widetilde{Q}^*)^2] = 0$. Taking $g = \mathbf{1}_A - \mathbb{E}[\mathbf{1}_A k_0]/\mathbb{E}[k_0]$ for any event A determined by H_0 shows that $r_0/(\widetilde{Q}^*)^2$ is proportional to k_0. Thus, for some $\lambda > 0$,

$$r_0 = \lambda(\widetilde{Q}^*)^2 k_0 \quad \text{almost surely.} \tag{18}$$

For comparison, uniform random sampling starts every task with probability $q_{\mathrm{unif}} = \bar{B}/\mathbb{E}[k_0]$. Assumption 3 implies $\epsilon < q_{\mathrm{unif}} < 1$. Equation (18) shows that $\widetilde{Q}^*$ is constant only when r_0/k_0 is constant almost surely. Therefore, if r_0/k_0 varies across tasks, strict convexity of the objective in Equation (17) implies that $\widetilde{Q}^*$ has strictly smaller variance than uniform random sampling at the same expected cost.

Global optimality when the criterion fails. For a general policy, define the inverse-probability increments

$$z_s = \frac{1}{Q(\mathcal{H}_s)} - \frac{1}{Q(\mathcal{H}_{s-1})} \geq 0, \qquad \frac{1}{Q(\mathcal{H}_{j-1})} = 1 + \sum_{s<j} z_s \leq \epsilon^{-1}. \tag{19}$$

Each z_s is measurable from $\mathcal{H}_s$. Their structural constraints are convex. Define the added score variance and expected cost per task by

$$D(z) = \mathbb{E}\left[\sum_{s<T} r_s z_s\right], \qquad K(z) = \mathbb{E}\left[\sum_{j=1}^{T} \frac{C_j}{1 + \sum_{s<j} z_s}\right]. \tag{20}$$

The tower property gives the expression for D because z_s is measurable from the same prefix used to define r_s. Thus D is linear and K is convex. The reference policy has $z_0^* = (\widetilde{Q}^*)^{-1} - 1$ and $z_s^* = 0$ for $s \geq 1$.

The tangent inequality for $x \mapsto 1/x$ gives

$$\begin{aligned} K(z) - K(z^*) &\geq -\mathbb{E}\left[(\widetilde{Q}^*)^2 \sum_{j=1}^{T} C_j \sum_{s<j} (z_s - z_s^*)\right] \\ &= -\mathbb{E}\left[(\widetilde{Q}^*)^2 \sum_{s<T} k_s (z_s - z_s^*)\right]. \end{aligned}$$

The exchange is valid because the sum of absolute increment changes on each path is at most $2(\epsilon^{-1} - 1)$ and total cost is integrable. Consequently,

$$D(z) - D(z^*) + \lambda\{K(z) - K(z^*)\} \geq \mathbb{E}\left[\sum_{s<T} \{r_s - \lambda(\widetilde{Q}^*)^2 k_s\}(z_s - z_s^*)\right]. \tag{21}$$

If Equation (10) fails almost surely at every nonterminal stage $s \geq 1$, its corresponding coefficients on the right are nonnegative by Equation (18). The coefficient at $s = 0$ is zero, and $z_s - z_s^* = z_s \geq 0$ for $s \geq 1$. Thus the right side is nonnegative. Every budget-feasible competitor has $K(z) \leq K(z^*) = \bar{B}$, which implies $D(z) \geq D(z^*)$.

A strict improvement when the criterion holds. Suppose some integer $s \geq 1$ admits an event $A \in \sigma(\mathcal{H}_s)$ of positive probability, contained in $\{s < T\}$, on which

$$r_s - r_0 k_s / k_0 < 0. \tag{22}$$

Set products with $\mathbf{1}_A$ to zero off $\{s < T\}$. For $h > 0$, increase the initial sampling probability and reduce continuation on A by setting

$$\begin{aligned} \widetilde{Q}_h(H_0) &= \widetilde{Q}^*(H_0) + h\,\frac{\mathbb{E}[\mathbf{1}_A k_s \mid H_0]}{k_0}, \\ Q^{(h)}(\mathcal{H}_t) &= \begin{cases} \widetilde{Q}_h(H_0), & 0 \leq t < s, \\ \widetilde{Q}_h(H_0) - h\mathbf{1}_A, & s \leq t < T. \end{cases} \end{aligned} \tag{23}$$

Also set $Q^{(h)}(\varnothing) = 1$. The event A is used only after stage s has been reached, so the policy depends only on observed information. Nonnegative costs and the tower property give $0 \leq \mathbb{E}[\mathbf{1}_A k_s \mid H_0] \leq k_0$. For $0 < h < \delta$, Assumption 3 therefore ensures $\epsilon \leq \widetilde{Q}_h - h \leq \widetilde{Q}_h \leq 1$. The policy is feasible for the structural constraints, and its expected cost is exactly

$$K(z(h)) = \mathbb{E}[\widetilde{Q}_h k_0] - h\mathbb{E}[\mathbf{1}_A k_s] = \mathbb{E}[\widetilde{Q}^* k_0] = \bar{B},$$

where $z(h)$ denotes the increments of $Q^{(h)}$.

Only the initial increment and the increment at stage s can be nonzero, so

$$D(z(h)) = \mathbb{E}[(\widetilde{Q}_h^{-1} - 1) r_0] + \mathbb{E}\left[\mathbf{1}_A \left\{\frac{1}{\widetilde{Q}_h - h} - \frac{1}{\widetilde{Q}_h}\right\} r_s\right].$$

The probability floor and bounded risks justify differentiation under the expectation. Equation (18) then gives

$$\begin{aligned}\left.\frac{d}{dh}D(z(h))\right|_{h=0} &= -\mathbb{E}\left[\frac{r_0}{(\widetilde{Q}^*)^2}\frac{\mathbb{E}[\mathbf{1}_A k_s \mid H_0]}{k_0}\right] + \mathbb{E}\left[\mathbf{1}_A \frac{r_s}{(\widetilde{Q}^*)^2}\right] \\ &= \mathbb{E}\left[\mathbf{1}_A \left\{\frac{r_s}{(\widetilde{Q}^*)^2} - \lambda k_s\right\}\right] < 0.\end{aligned} \tag{24}$$

The integrand is integrable because $\widetilde{Q}^* \geq \epsilon$, $r_s \leq 1$, and $\mathbb{E}[\mathbf{1}_A k_s] \leq \mathbb{E}[k_0] < \infty$. Hence a sufficiently small positive h strictly reduces variance at exactly the same expected cost. The policy stops a started run with positive probability because $\Pr(A) > 0$ and $\widetilde{Q}_h \geq \epsilon$. This completes both directions of the proposition. □