MOUSE: Precision File-Editing Tools for AI Coding Agents

A Controlled Study of Tool Architecture Effects on Agent Performance

Technical Report with Complete Statistical Analysis

Simon W. Reiff

Working Draft — January 2026

Abstract

AI coding agents increasingly rely on file-editing operations, yet standard editing paradigms require agents to echo file content in responses—creating inefficiencies in cost, time, and reliability. We present MOUSE, a suite of precision file-editing tools for AI coding agents exposed via the Model Context Protocol. Across three preregistered confirmatory studies ($N = 67$ paired trials), we compared MOUSE-enabled GitHub Copilot instances against baseline instances using standard editing tools, both running in isolated Docker containers on timed file-editing benchmarks.

Results demonstrate statistically significant advantages for MOUSE:

- **Precision**: 56% vs 0% first-try correctness (risk difference = +56 percentage points, 95% CI [+33, +73] pp, McNemar two-sided $p = 1.22 \times 10^{-4}$)
- **Capability**: 89.5% vs 0% success on structured data manipulation (risk difference = +89.5 pp, 95% CI [+63, +97] pp, McNemar one-sided $p = 7.63 \times 10^{-6}$)
- **Efficiency**: On tasks it can complete, Baseline costs 58% more (geometric mean ratio $G = 1.58$, 95% CI $[1.35, 1.91]$, permutation one-sided $p = 7.15 \times 10^{-7}$) and takes 3.6× longer (95% CI $[2.77, 4.06]\times$, permutation one-sided $p = 1.19 \times 10^{-7}$)

All primary hypotheses achieved statistical significance at preregistered $\alpha = 0.05$ thresholds, with effect sizes ranging from "large" to "very large" by conventional criteria. Our benchmark methodology isolates tool architecture as an independent variable while holding the model constant—an approach we believe is novel in AI agent evaluation. These findings suggest that tool architecture, not just model capability, is a critical and underexplored lever for AI coding agent performance. Limitations include benchmark specificity and single-baseline comparison.

Keywords: AI coding agents, developer tools, file editing, large language models, tool use, Model Context Protocol

Contents

1 Introduction

1.1 The File-Editing Bottleneck

Large language models (LLMs) have transformed software development through AI coding assistants that can generate, explain, and modify code [1, 2]. Tools like GitHub Copilot, Cursor, and Claude integrate directly into development environments, promising to accelerate engineering workflows. However, a fundamental bottleneck constrains their effectiveness: **file editing**.

When an AI agent needs to modify existing code, standard approaches require the agent to:

1. Read the relevant file content
2. Identify the section to change
3. Output both the old content (to locate the edit) and new content (to replace it)

This "oldString/newString" paradigm forces agents to **echo substantial portions of files in their responses**. A simple deletion of 50 lines requires transmitting those 50 lines back to the system—even though the intent could be expressed as "delete lines 10–60."

1.2 The Cost of Verbosity

The economic implications are significant. Output tokens typically cost 3–5× more than input tokens in commercial LLM APIs [3]. When agents must echo file content, they:

- **Waste tokens** on content that already exists in the system
- **Increase latency** as longer responses take more time to generate
- **Risk errors** when reproducing whitespace, indentation, or special characters

For tasks involving large files or repetitive edits, these inefficiencies compound.

1.3 Research Questions

RQ1 (Precision) Does Mouse improve first-try correctness compared to Agents using built-in find-and-replace tools?

RQ2 (Capability) Does Mouse unlock tasks that are currently infeasible for AI Agents to perform under equivalent resources?

RQ3 (Speed) Is Mouse faster than built-in tools at performing equivalent work?

RQ4 (Cost) Is Mouse less expensive than built-in tools at performing equivalent work?

1.4 Contributions

This paper makes three contributions:

1. **Mouse**, a toolkit of precision file-editing operations designed for AI coding agents, demonstrating that declarative, compact commands can outperform verbose editing paradigms.
2. **Empirical evidence** from three preregistered confirmatory studies ($N = 67$ pairs) using a novel benchmark methodology—the first to isolate AI agent file-editing tool architecture as an independent variable while holding the model constant.
3. **30 numbered findings** with effect sizes, confidence intervals, and explicit test directionality, enabling independent verification.

2 Background and Related Work

2.1 AI Coding Assistants

The landscape of AI coding assistants has expanded rapidly since GitHub Copilot's introduction in 2021 [1]. Current tools span from code completion (Copilot, Tabnine) to full agentic systems (Cursor Composer [2], Cline, Aider). These tools increasingly integrate with development environments to perform file operations, execute tests, and iterate on solutions.

2.2 Tool Use in Large Language Models

Modern LLMs support "function calling" or "tool use"—invoking external functions and incorporating results [3, 10]. The Model Context Protocol (MCP) [5] provides a standardized interface for LLM-tool communication. Recent evaluations of tool-augmented agents include ToolBench [11], which assesses function-calling accuracy but not editing-tool design. However, the *design* of these tools remains underexplored; most research focuses on model capability rather than tool architecture.

2.3 Programming by Edit and Patch-Based Approaches

Prior work on structured editing includes syntax-directed editors [6]. Recent LLM code editing approaches include Aider's "architect" mode [12]. Our work differs in focusing on the *interface* between LLM agents and editing operations, though MOUSE also contributes novel editing algorithms.

2.4 Evaluation Benchmarks

Existing benchmarks for AI coding agents can be organized into three categories, each evaluating a different aspect of agent performance. We identify a fourth category—AI agent file-editing tool benchmarks—that our work introduces.

2.4.1 Agent-Level Benchmarks

These benchmarks evaluate end-to-end agent performance on coding tasks, measuring whether agents can solve problems correctly. SWE-bench [4] evaluates agents on real GitHub issues, requiring them to produce patches that pass repository test suites. HumanEval [13] and MBPP [14] assess code generation from natural language specifications. APPS [15] provides competitive programming problems at varying difficulty levels. In all cases, the evaluation target is the agent's reasoning and code generation capability; the editing tools are assumed to be sufficient and are not varied.

2.4.2 Tool-Use and Function-Calling Benchmarks

A second category evaluates how well agents select and invoke tools from a provided set. API-Bank [19] assesses agents on multi-step API call sequences. ToolBench [11] provides thousands of real-world APIs for tool selection evaluation. Gorilla [20] tests API call accuracy against massive API corpora. The Berkeley Function Calling Leaderboard (BFCL) [21] provides standardized function-calling evaluations. These benchmarks test whether agents can correctly *invoke* tools, but not whether the tools themselves are well-designed for agent use.

2.4.3 Edit Generation Quality Benchmarks

A third category evaluates whether agents can produce correct code edits. CanItEdit [16] tests whether LLMs can follow code editing instructions, measuring the quality of generated patches.

CodeEditorBench [17] evaluates edit generation across multiple programming languages. EditBench [18] provides realistic instructed code editing tasks derived from real development scenarios. These benchmarks assume the edits are applied correctly by the environment; they do not evaluate the edit *application* mechanism itself.

2.4.4 The AI Agent File-Editing Tool Gap

All three categories share a common assumption: the file-editing interface is held constant while agent capability, tool selection accuracy, or edit generation quality is varied. None isolates the *editing tool architecture* as an independent variable.

To our knowledge, no prior benchmark family has been designed to evaluate AI agent file-editing tools while holding the agent model constant. The BX-50X and BX-70X families of benchmarks introduced in this paper fill this gap: we fix the model (Claude Sonnet 4.5 or Haiku 4.5), fix the task, and vary only the editing tool interface. This enables controlled comparison of editing paradigms—a methodological contribution complementary to existing benchmark categories.

2.5 The File-Editing Problem

Existing approaches to file editing fall into two categories:

Find-and-Replace The dominant paradigm, requiring agents to specify both old content (to locate) and new content (to substitute). Examples include VS Code's `replace_string_in_file` and `multi_replace_string_in_file`.

Diff-Based Some systems use unified diff format, but this still requires outputting deleted lines with "-" prefixes—eliminating potential efficiency gains.

Neither approach addresses the fundamental problem: agents must echo content that already exists.

3 The Mouse Approach

Mouse replaces the standard "oldString/newString" editing paradigm with a suite of compact, declarative operations exposed via MCP. Rather than requiring AI agents to echo file content, Mouse allows agents to express editing intent directly.

3.1 High-Level Architecture

Mouse provides AI agents with editing primitives that:

- **Address content by position** rather than requiring content matching
- **Express operations declaratively** (e.g., "delete lines 10–50")
- **Execute atomically** with built-in staging and rollback mechanisms

3.2 Key Capabilities

The Mouse toolkit (version 0.9.x) provides 10 tools spanning:

- Line-based insertions and deletions
- Literal text replacement
- Bulk operations across line ranges

- Structured content relocation
- Staged editing with explicit save/cancel

Implementation details are proprietary; this paper reports only observable behavioral differences.

3.3 Comparison to Baseline

Where baseline tools require outputting both the text being replaced *and* the replacement, MOUSE operations typically require only replacement content (for modifications) or positional parameters (for deletions). This architectural difference is the hypothesized mechanism for the efficiency and reliability gains reported in Section 5.

4 Methods

4.1 Study Design

We conducted three independent confirmatory studies using a **within-subjects paired design**:

- Preregistered hypotheses, analysis plans, and stopping rules before data collection
- Paired randomization (MOUSE-first vs Baseline-first balanced via PRNG)
- Model version, temperature, and task content held constant across conditions

4.2 Experimental Environment

Both conditions used GitHub Copilot (Claude Sonnet 4.5 or Haiku 4.5, depending on study) running in VS Code within isolated Docker containers. The only difference was MOUSE tool availability.

Table 1: Container environment configuration.[1]

Component	Baseline	Mouse
Base Image	code-server:latest	code-server:latest
Runtime	Node.js 20.x	Node.js 20.x
Editor	VS Code + Copilot	VS Code + Copilot
Editing Tools	`replace_string_in_file`	MOUSE v0.9.x (10 tools)

4.3 Cost Measurement

Cost definition: Costs reported in this paper are **imputed API costs** calculated from measured token counts using Anthropic's public rate card. We used the rate card in effect during data collection (December 2025): Claude Sonnet 4.5 at $3/M input ($0.30/M cached), $15/M output; Claude Haiku 4.5 at $1/M input ($0.10/M cached), $5/M output.[2] Token counts were captured from the telemetry stream and include input tokens (system prompt, context, tool results) and output tokens (model responses, tool calls). GitHub Copilot does not expose per-request billing; these imputed costs represent what equivalent direct API usage would cost and enable cross-condition comparison. We additionally report **pricing-agnostic token deltas** (Tables 27, 28, 29) to enable cost re-estimation under alternative rate cards.

[1] For each instance, the specific versions of VS Code, GitHub Copilot, Copilot Chat, Node.js, and Mouse were pinned and recorded.

[2] Rate card retrieved from `https://claude.com/platform/api` on December 15, 2025. API pricing is subject to change; readers should verify current rates for cost projections.

4.4 Instruction Neutrality

Task instructions were designed to be tool-agnostic. Instructions did not direct agents to use any particular tool, operation, or strategy. Where general guidance was provided (e.g., "consider batching operations"), it was given equally to both conditions. Both conditions received structurally parallel instruction files differing only in tool-specific quick references.

4.5 The Three Studies

Table 2: Summary of the three confirmatory studies.

Study	Version	N	Task	Model	τ
BX-504B	v0.9.5	25 pairs	Multi-block refactoring	Haiku 4.5	180s
BX-701R	v0.9.7	19 pairs	CSV column relocation	Sonnet 4.5	240s
BX-504D	v0.9.7	23 pairs	Legacy code deletion	Haiku 4.5	120s

4.5.1 Study BX-504B: Precision Editing

BX-504B is a single-file surgical editing task on `data-transformer.js` (445 → 158 lines). The agent must delete legacy code blocks, delete legacy imports, and uncomment modern code blocks—all marked with specific comment delimiters. Success requires byte-for-byte match to the answer key.

Preregistered Endpoints:

- **Primary (H1)**: Perfect First Try (PFT) rate—exact match on first attempt
- **Secondary (H2)**: Success rate within $\tau = 180$s (gated by H1)
- **Secondary (H3)**: Cost Per Success (gated by H2)

4.5.2 Study BX-701R: Capability Boundary

The "Evil Square CSV" task requires relocating a column across 126 rows of structured data. This task stresses both context limits (agents must reference many rows) and output limits (baseline tools must echo substantial content). This study tests whether Mouse's compact notation can succeed where verbose approaches fail under equivalent time and resource constraints.

Preregistered Endpoints:

- **Primary (H1)**: Truncated Time-to-Success (T_τ) with $\tau = 240$s
- **Secondary (H2)**: Success rate (gated by H1)

4.5.3 Study BX-504D: Economic Efficiency

The BX-504D benchmark presents a 594-line legacy source file that must be completely deleted and replaced with a one-line deprecation comment followed by a newline. This simpler deletion task is designed to measure cost and time efficiency when both conditions can succeed at high rates, enabling direct economic comparison without capability-boundary confounds.

Preregistered Endpoints:

- **Primary (H1)**: Geometric Mean Paired Cost Ratio (G)
- **Secondary (H2)**: Time efficiency (gated by H1)

4.6 Metrics

Perfect First Try (PFT) Task correct on first attempt, before any feedback or retry

Success Rate Task completed with 100% accuracy within timeout τ

Truncated Time (T_τ) Completion time if successful, or timeout τ if not

RMTS Restricted Mean Time Saved: mean of paired differences ($T_{\tau,\text{Baseline}} - T_{\tau,\text{Mouse}}$)

Cost Imputed API cost (input + output tokens at rate card prices)

Cost Ratio (G) Geometric mean of paired cost ratios: $G = \exp(\text{mean}(\log(R_i)))$ where $R_i = \text{Cost}_{\text{Baseline},i}/\text{Cost}_{\text{Mouse},i}$

4.7 Statistical Methods

Our evaluation employs statistical methods that are standard in clinical trials and experimental psychology—paired designs, exact tests for small samples, effect sizes with confidence intervals—but which we believe warrant explicit presentation given variable statistical reporting practices across research communities. We describe our choices in detail to facilitate replication and comparison.

4.7.1 Test Directionality

Test directionality was preregistered separately for each study based on calibration data and theoretical expectations:

- **BX-504B**: Two-sided McNemar tests (H1, H2) at $\alpha = 0.05$; one-sided permutation test for cost (H3)
- **BX-701R**: One-sided tests throughout at $\alpha = 0.05$ (justified by strong directional calibration: 8/8 discordant pairs favored MOUSE in pilot)
- **BX-504D**: One-sided tests throughout at $\alpha = 0.05$ (directional hypotheses preregistered)

All p-values reported in this paper indicate their directionality explicitly.

4.7.2 Primary Tests

- **Binary outcomes (PFT, Success)**: Exact McNemar test on the 2×2 discordance table
- **Continuous outcomes (Cost, Time)**: Exact paired permutation test (2^N enumeration when $N \leq 23$; 10,000 random permutations otherwise)

4.7.3 Effect Sizes

We report multiple complementary effect size measures to characterize the magnitude and practical significance of observed differences.

Risk Difference (Primary for Binary Outcomes). The risk difference $\text{RD} = p_M - p_B$ provides an absolute measure of the probability improvement from MOUSE over Baseline. This metric handles boundary values (0% or 100%) naturally and has direct practical interpretation: "Mouse succeeds |RD| percentage points more often than Baseline."

Number Needed to Treat (NNT). The NNT quantifies how many tasks must use MOUSE instead of Baseline to gain one additional success:

$$\text{NNT} = \frac{1}{|\text{RD}|}$$

For example, NNT = 1.8 means switching 1.8 tasks from Baseline to MOUSE yields one additional first-try success on average.

Conditional Win Rate. For paired designs, the conditional win rate $w = b/(b+c)$ gives the proportion of discordant pairs won by MOUSE, where b = pairs MOUSE won and c = pairs Baseline won. This metric directly addresses: "When the conditions disagree, how often does MOUSE win?"

Cohen's h (Binary Outcomes, Post-Hoc). For proportion differences, we compute Cohen's h using the arcsine transformation:

$$h = 2\arcsin\sqrt{p_M} - 2\arcsin\sqrt{p_B}$$

Conventional thresholds: $|h| \geq 0.2$ (small), $|h| \geq 0.5$ (medium), $|h| \geq 0.8$ (large).

Paired Cohen's d_z (Continuous Outcomes, Preregistered for BX-504D). For continuous metrics (time, cost) in paired designs, we compute the standardized mean difference using the standard deviation of the paired differences:

$$d_z = \frac{\bar{D}}{s_D}, \quad \bar{D} = \frac{1}{n}\sum_{i=1}^{n}(x_{B,i} - x_{M,i}), \quad s_D = \sqrt{\frac{\sum_{i=1}^{n}(D_i - \bar{D})^2}{n-1}}$$

where $D_i = x_{B,i} - x_{M,i}$ is the paired difference for the i-th task. This paired d_z (rather than the independent-groups pooled-SD formula) is the appropriate effect size for within-subject designs [25]. Conventional thresholds: $|d_z| \geq 0.2$ (small), $|d_z| \geq 0.5$ (medium), $|d_z| \geq 0.8$ (large).

Odds Ratio (Supplementary). When cells contain zeros, we apply the Haldane-Anscombe correction (adding 0.5 to each cell) to obtain finite estimates:

$$\text{OR}_{\text{HA}} = \frac{(a+0.5)(d+0.5)}{(b+0.5)(c+0.5)}$$

where a, b, c, d are the 2×2 table cell counts. We emphasize that OR is supplementary to risk difference when zero cells are present, as CIs become extremely wide.

4.7.4 Confidence Intervals

All 95% confidence intervals are two-sided unless otherwise noted. We employ methodology appropriate to each quantity type.

Risk Difference CIs (Paired Binary Outcomes). For paired binary outcomes (e.g., PFT, success rates), the paired risk difference is $\widehat{RD} = (b-c)/n$ where b and c are the discordant counts from the 2×2 paired table (see Section 4.7.2). We employ **Tango's score-based confidence interval** for paired proportion differences [26], which provides excellent coverage properties for paired designs even with boundary values. The interval is derived from:

$$\widehat{RD} = \frac{b-c}{n}, \quad \text{SE}(\widehat{RD}) = \frac{\sqrt{b+c-(b-c)^2/n}}{n}$$

with score-based adjustment for small samples. This paired-appropriate method (rather than the independent-proportions Newcombe interval) correctly accounts for within-pair correlation.

Effect Size CIs for Cohen's h. For the arcsine-transformed effect size $h = 2\arcsin\sqrt{p_1} - 2\arcsin\sqrt{p_2}$, we derive CIs via the arcsine transformation standard error [8]:

$$\mathrm{SE}(h) = \sqrt{\frac{1}{n_1} + \frac{1}{n_2}}$$

yielding $\mathrm{CI}_{95} = h \pm 1.96 \cdot \mathrm{SE}(h)$.

Effect Size CIs for Paired d_z and Hedges' g. For continuous outcomes (e.g., completion time, token cost) in paired designs, we report both paired d_z and the bias-corrected Hedges' g. The CI for d_z is computed via **bootstrap percentile intervals** with $B = 10{,}000$ resamples (seed = 20260107), which is robust to non-normality in the difference distribution. The Hedges' g correction factor is:

$$g = d_z \cdot J(n-1), \quad J(m) = \frac{\Gamma(m/2)}{\sqrt{m/2}\,\Gamma((m-1)/2)} \approx 1 - \frac{3}{4m-1}$$

Note: For paired designs, the degrees of freedom for $J(\cdot)$ is $n-1$ (not n_1+n_2-2 as in independent designs) [25].

Ratio CIs (Speedup, Cost Ratio). For ratio metrics (e.g., time speedup, geometric mean cost ratio G), we employ **bootstrap percentile intervals** with $B = 10{,}000$ resamples (seed = 20260107 for reproducibility). The bootstrap distribution is constructed by resampling paired observations with replacement, computing the ratio for each resample, and extracting the 2.5th and 97.5th percentiles.

Variance Ratio CIs. For variance ratios (comparing dispersion between conditions), we use the F-distribution approximation:

$$\mathrm{CI}_{95} = \left[\frac{s_1^2/s_2^2}{F_{n_1-1,n_2-1,0.975}},\ \frac{s_1^2/s_2^2}{F_{n_1-1,n_2-1,0.025}}\right]$$

A_{12} (Vargha-Delaney) CIs. For the stochastic superiority measure A_{12}, we employ bootstrap percentile intervals with 10,000 resamples, as this nonparametric effect size lacks a closed-form sampling distribution.

4.7.5 Bayesian Analysis

For binary outcomes, we computed Beta-binomial posteriors to estimate $P(\pi_M > \pi_B \mid \text{data})$—the probability that MOUSE's true success rate exceeds Baseline's, given the observed data.

Model Specification.

- **Likelihood**: $k_M \sim \text{Binomial}(n_M, \pi_M)$ and $k_B \sim \text{Binomial}(n_B, \pi_B)$, independently.
- **Prior**: $\pi_M, \pi_B \sim \text{Beta}(1, 1)$ (uniform), representing no prior preference.
- **Posterior**: By conjugacy, $\pi_M \mid k_M \sim \text{Beta}(k_M + 1, n_M - k_M + 1)$ and similarly for π_B.

Probability of Superiority. The quantity of interest is:

$$P(\pi_M > \pi_B \mid \text{data}) = \int_0^1 \int_0^{\pi_M} f_M(\pi_M) \cdot f_B(\pi_B)\, d\pi_B\, d\pi_M$$

where f_M and f_B are the respective Beta posterior densities. We evaluate this integral via Monte Carlo simulation: draw $S = 100{,}000$ samples $(\pi_M^{(s)}, \pi_B^{(s)})$ from the joint posterior, then compute

$$\widehat{P}(\pi_M > \pi_B) = \frac{1}{S} \sum_{s=1}^{S} \mathbf{1}\{\pi_M^{(s)} > \pi_B^{(s)}\}$$

Interpretation. Unlike frequentist p-values (which quantify evidence against a null), the Bayesian posterior probability directly answers: "Given our data and prior beliefs, how confident should we be that MOUSE is genuinely better?" Values exceeding 0.95 indicate strong evidence of superiority; values exceeding 0.99 indicate very strong evidence. This Bayesian supplement provides a probability-of-superiority interpretation alongside the frequentist p-values.

Limitation: Independence Assumption. This Beta-Binomial model treats marginal success rates as independent and does not explicitly model within-pair correlation. The paired structure of our design (which induces positive correlation between conditions on the same task) is instead captured by the frequentist McNemar and permutation tests. The Bayesian posterior should be interpreted as a descriptive supplement for probability-of-superiority, not as a replacement for the paired-design inference.

4.8 Methods Rationale

Agent runs exhibit substantial between-instance variance due to stochastic sampling and API-level non-determinism. **Pairing**—running both conditions on identical benchmark instances—controls this variance and isolates the tool-architecture effect. **Randomization-based inference** (permutation tests, exact binomial) avoids fragile distributional assumptions that may not hold for small-to-moderate paired samples. This design follows recent recommendations for more rigorous LLM evaluation methodology [22].

We present standard effect summaries (risk differences, ratios, 95% CIs) as the primary inferential language, with exact enumeration and distribution-free bounds as robustness checks. Cross-check statistics (sign tests, Wilcoxon) confirm that conclusions are not sensitive to analytical choices.

4.9 Validity Checks

A/A Validation A Baseline-vs-Baseline study (N = 5 pairs) confirmed no systematic arm bias in the telemetry harness (identical success rates, Fisher $p = 1.0$).

Order Effects Linear regression on cost ratio vs execution order found $|\beta| < 0.03$, $|r| < 0.03$ (BX-504D), confirming no learning or fatigue confounds.

Session Effects No time-of-day or session-number confounds detected ($|r| < 0.2$ for all).

Randomization Mulberry32 PRNG with preregistered seeds determined within-pair order and session assignment.

5 Results

We organize results into two parts to distinguish claims with different epistemic status.

Part A: Confirmatory Findings. Findings corresponding to *preregistered primary and secondary hypotheses* with $\alpha = 0.05$ error-rate control. Per our protocols, secondary hypotheses were tested only if the primary hypothesis was significant (hierarchical gatekeeping). We report *all* preregistered results—including null findings—to ensure complete transparency.

Part B: Robustness Analyses and Effect Characterization. Effect size characterizations, cross-check statistics, robustness bounds, and mechanism explorations. These supplement the confirmatory findings and demonstrate that conclusions are not sensitive to analytical choices. No additional familywise error claims are made. Detailed exploratory analyses (PCA, K-S tests, shift functions) appear in Appendix B.

5.1 Part A: Confirmatory Findings

The following findings correspond to preregistered hypotheses tested at $\alpha = 0.05$.

5.1.1 Study BX-504B: Primary Hypothesis (H1)

Study BX-504B evaluated precision editing performance using Claude Haiku 4.5 with a 180-second timeout and two-sided hypothesis tests.

Finding A-1: Mouse Achieves 56% First-Try Correctness; Baseline Achieves 0%

Table 3: Perfect First Try (PFT) results from Study BX-504B ($N = 25$ pairs).

Metric	Mouse	Baseline	Statistical Test
PFT Rate	14/25 (56.0%)	0/25 (0.0%)	McNemar exact (two-sided)
95% CI (Wilson)	[37.1%, 73.3%]	[0.0%, 13.3%]	$p = 1.22 \times 10^{-4}$

The McNemar 2×2 discordance table:

Table 4: McNemar discordance table for PFT (Study BX-504B).

	Baseline PFT=1	Baseline PFT=0
Mouse PFT=1	0	14
Mouse PFT=0	0	11

Discordant pairs: $b + c = 14 + 0 = 14$. The two-sided exact McNemar p-value is $2 \times 0.5^{14} = 1.22 \times 10^{-4}$.

Cross-check (sign test): In 14 of 14 discordant pairs, MOUSE achieved PFT while Baseline did not. Under the null hypothesis of equal probability, this is equivalent to observing 14 heads in 14 fair coin flips (sign test $p = 6.1 \times 10^{-5}$ one-sided, 1.22×10^{-4} two-sided), confirming the McNemar result.

Interpretation: On a precision code-editing task, MOUSE achieved first-try correctness in 56% of attempts. Baseline achieved 0% across 25 independent attempts under identical conditions.

5.1.2 Study BX-504B: Secondary Hypotheses (H2, H3)

Finding A-2: BX-504B H2 (Success Rate) Did Not Reach Statistical Significance

Per our preregistered protocol, we tested H2 (Success Rate) only because H1 was significant.

Table 5: Success rate results from Study BX-504B ($N = 25$ pairs). **Not statistically significant.**

Metric	Mouse	Baseline	Statistical Test
Success Rate	19/25 (76.0%)	13/25 (52.0%)	McNemar exact (two-sided)
Discordant pairs	$b = 9$, $c = 3$		$p = 0.146$

Result: The two-sided McNemar p-value of 0.146 did not reach statistical significance at $\alpha = 0.05$. While the point estimate favors MOUSE (76% vs 52%), we cannot reject the null hypothesis that the success rates are equal.

Interpretation: The PFT advantage (Finding A-1) does not translate to a statistically significant success rate advantage in this sample. The 10 concordant successes reduced effective sample size for the McNemar test to 12 discordant pairs.

Finding A-3: BX-504B H3 (Cost Per Success) Was Gated and Not Tested

Per our preregistered hierarchical gatekeeping procedure, H3 was **not tested** because H2 did not reach significance. This maintains family-wise error control but precludes formal cost-efficiency claims from Study BX-504B.

5.1.3 Study BX-504B: Per-Protocol Sensitivity Analysis

Our preregistered protocol specified a per-protocol (PP) sensitivity analysis excluding runs with documented rate-throttling indicators. Four pairs exhibited rate-throttling telemetry ("rate limit exceeded" errors in API logs), leaving N=21 pairs for PP analysis.

Finding A-4: PP Analysis Shows Significant H2 and H3 Results

Table 6: Per-Protocol sensitivity analysis results (N=21 pairs). **Significant but exploratory.**

Hypothesis	Metric	Result	McNemar p
H2	Success Rate	17/21 (81.0%) vs 10/21 (47.6%)	0.021
H3	Cost Per Success	MOUSE lower in 10/10 discordant	0.021

Interpretation: When rate-throttling noise is removed, both H2 and H3 reach significance at $\alpha = 0.05$. However, because the ITT analysis governs our confirmatory claims, these PP results are properly classified as *preregistered sensitivity analyses*—they inform interpretation but do not override the null ITT finding for H2.

Epistemic status: The PP analysis was preregistered and follows the gatekeeping hierarchy (H2 PP significant ⇒ test H3 PP). We report it for completeness and to illustrate that measurement noise (rate-throttling) likely attenuated the ITT effect. Readers should weight the ITT result (Finding A-2) as primary.

5.1.4 Study BX-701R: Primary and Secondary Hypotheses

Study BX-701R evaluated capability boundaries using Claude Sonnet 4.5 with a 240-second timeout on a 126-row column-relocation task designed to stress context and output limits. All tests were preregistered as one-sided based on strong directional calibration evidence.

Finding A-5: BX-701R H1 (T_τ) and H2 (Success) Both Significant

Table 7: Confirmatory results from Study BX-701R ($N = 19$ pairs).

Hyp.	Metric	Result	p-value
H1	T_τ (cont.)	RMTS = 169.2s (MOUSE 70.8s vs Baseline 240.0s)	7.63×10^{-6}
H2	Success (binary)	89.5% vs 0% (17/19 vs 0/19)	7.63×10^{-6}

T_τ calculation: The truncated time-to-success T_τ assigns each run its completion time if successful, or the timeout $\tau = 240$s if not. For MOUSE: mean $T_\tau = 70.8$s (17 successes with times ranging from 14.7s to 177.1s, plus 2 timeouts at 240s). For Baseline: mean $T_\tau = 240.0$s (all 19 runs timed out). The Restricted Mean Time Saved (RMTS) is the mean of paired differences:

$$\text{RMTS} = \frac{1}{n}\sum_{i=1}^{n}(T_{\tau,B,i} - T_{\tau,M,i}) = 240.0 - 70.8 = 169.2\text{s}$$

A 95% bootstrap CI for RMTS (10,000 resamples, BCa method) yields $[128.3, 194.1]$s, confirming substantial time savings even accounting for sampling variability.

H1 p-value derivation (permutation test): For continuous T_τ, we use an exact paired permutation test. Under H_0, the sign of each paired difference $D_i = T_{\tau,B,i} - T_{\tau,M,i}$ is exchangeable. With $n = 19$ pairs, there are $2^{19} = 524{,}288$ possible sign permutations. Of these, only 4 yield a mean difference $\geq$ observed (169.2s)—corresponding to the $2^2 = 4$ equivalent sign patterns from the 2 tie pairs where both conditions timed out. Thus $p = 4/524{,}288 = 7.63 \times 10^{-6}$.

H2 p-value derivation (McNemar): For binary success, we use the exact McNemar test. With 17 discordant pairs all favoring MOUSE ($b = 17$, $c = 0$), the one-sided p-value is $P(X \geq 17 \mid X \sim \text{Binomial}(17, 0.5)) = 0.5^{17} = 7.63 \times 10^{-6}$.

Note on p-value coincidence: The identical p-values for H1 and H2 are a mathematical coincidence arising from the data structure: (1) the permutation test's 4 extreme permutations come from 2^2 (two tie pairs), yielding $4/2^{19}$; (2) McNemar's 0.5^{17} happens to equal this same value. Both tests are correctly specified for their respective outcome types (continuous vs binary).

Interpretation: For this column-relocation task under the tested tool interface and 240s timeout, Baseline achieved 0/19 completions while MOUSE achieved 17/19. Both preregistered hypotheses are confirmed. This represents a capability boundary observed under these specific conditions; alternative baseline implementations or longer timeouts might yield different results.

5.1.5 Study BX-504D: Primary and Secondary Hypotheses

Study BX-504D evaluated economic efficiency using Claude Haiku 4.5 with a 120-second timeout on a file deprecation task. All tests were preregistered as one-sided.

Finding A-6: BX-504D H1 (Cost Ratio) Significant

Table 8: Cost efficiency confirmatory results from Study BX-504D ($N = 23$ pairs).

Metric	Value
Geometric Mean Cost Ratio G	1.5830
95% CI (bootstrap, 10,000 samples)	$[1.3453, 1.9118]$
95% CI (permutation-inversion)	$[1.3106, 1.9129]$
Permutation test (one-sided)	$p = 7.15 \times 10^{-7}$
Total permutations tested	$2^{23} = 8,388,608$

Interpretation: Baseline costs 58% more per task than MOUSE. The 95% CI lower bound (1.31×) provides a distribution-free floor under the randomization design: at least a 31% cost premium for Baseline.

Cross-check (Wilcoxon): Wilcoxon signed-rank test [29] on log-transformed cost ratios confirms the permutation result (W = 276, $p < 0.001$), demonstrating that the conclusion is robust to statistical method choice.

Finding A-7: BX-504D H2 (Time) Significant

Table 9: Time efficiency confirmatory results from Study BX-504D ($N = 23$ pairs).

Metric	Mouse	Baseline
Geometric Mean Time	12.48s	44.69s
Speedup Ratio	3.58×, 95% CI $[2.77, 4.06]$	
Permutation test (one-sided)	$p = 1.19 \times 10^{-7}$	

Geometric mean calculation: The geometric mean time for each condition is computed as:

$$\bar{T}_{\text{geom}} = \exp\left(\frac{1}{n}\sum_{i=1}^{n}\log T_i\right) = \left(\prod_{i=1}^{n} T_i\right)^{1/n}$$

For MOUSE ($n = 23$): $\bar{T}_{\text{geom,M}} = \exp(\text{mean}(\log T_{M,i})) = 12.48$s. For Baseline ($n = 23$): $\bar{T}_{\text{geom,B}} = \exp(\text{mean}(\log T_{B,i})) = 44.69$s. The speedup ratio is $44.69/12.48 = 3.58\times$.

Interpretation: MOUSE completes tasks 3.58× faster than Baseline. The permutation test confirms this is not due to chance ($p = 1.19 \times 10^{-7}$).

5.1.6 Summary: Confirmatory Results (H1 Confirmed in All Studies)

Table 10: Summary of confirmatory hypothesis tests across all three studies.

Study	Hypothesis	Endpoint	Result	p-value
BX-504B	H1	PFT Rate	**Confirmed**	1.22×10^{-4}
BX-504B	H2	Success Rate	Not significant	0.146
BX-504B	H3	Cost Per Success	Gated	—
BX-701R	H1	T_τ Completion	**Confirmed**	7.63×10^{-6}
BX-701R	H2	Success Rate	**Confirmed**	7.63×10^{-6}
BX-504D	H1	Cost Ratio	**Confirmed**	7.15×10^{-7}
BX-504D	H2	Time Ratio	**Confirmed**	1.19×10^{-7}

Six of seven preregistered hypotheses were confirmed at $\alpha = 0.05$. The one null result (BX-504B H2) is transparently reported, with its downstream hypothesis (H3) appropriately gated per the preregistered hierarchical procedure.

5.2 Part B: Robustness Analyses and Effect Characterization

The following findings provide robustness analyses, effect characterization, and cross-check statistics that supplement the confirmatory results above. They demonstrate that conclusions are robust to analytical choices and are not artifacts of specific statistical methods.

5.2.1 Effect Size Dashboard

Table 11 provides a consolidated view of effect sizes across all three studies. Effect sizes marked with † were preregistered; others are post-hoc characterizations.

Table 11: Effect size dashboard: Primary endpoints across studies.

Study	Endpoint	Effect Size	Value	Interpretation
Binary outcomes				
BX-504B	PFT Rate	Risk Diff.	+56 pp [+33, +73]	56% absolute improvement
		Cohen's h	1.69 [1.14, 2.25]	>2× "large"
BX-701R	Success	Risk Diff.	+89.5 pp [+63, +97]	89.5% absolute improvement
		Cohen's h	2.48 [1.84, 3.12]	>3× "large"
Continuous outcomes				
BX-504D	Time	$d_z^\dagger$	1.23 [1.09, 2.66]	>1.5× "large"
		$A_{12}^\dagger$	1.00 [1.00, 1.00]	Complete dominance
	Cost	$G^\dagger$ (ratio)	1.58× [1.35, 1.91]	Baseline 1.58× MOUSE
BX-701R	T_τ	$A_{12}^\dagger$	0.95	Near-complete dominance
Correlation (post-hoc)				
BX-701R	Success	ϕ (= r)	0.90 [0.81, 0.95]	81% outcome variance

Key finding: Across all three studies, effect sizes range from "large" to "very large" by conventional criteria. The consistency across multiple orthogonal measures—risk difference, standardized mean difference, probability of superiority, and correlation—provides strong evidence that the observed MOUSE advantages are not artifacts of any particular effect size measure.

5.2.2 Effect Size Characterization (BX-504B)

Finding B-1 (BX-504B): Risk Difference of +56 Percentage Points; OR = 64× (Supplementary)

Table 12: Effect sizes for precision (Finding B-1). Risk difference is primary; OR is supplementary.

Metric	Value	Note
Risk Difference	+56.0 pp, 95% CI [+33, +73] pp	Primary effect size
NNT	1.8	Tasks to switch for one additional PFT
Conditional Win Rate	14/14 = 100%	[76.8%, 100%] Clopper-Pearson
Cohen's h	1.69 [1.14, 2.25]	>2× "large" (0.8); post-hoc
Odds Ratio (Haldane)	64.3×	Suppl.; 0-cell corrected
OR 95% CI	[3.5, 1173]	Wide due to 0-cell correction

Note on OR: When Baseline has 0 successes, the odds ratio is technically infinite. The Haldane-Anscombe correction (adding 0.5 to each cell) yields a finite estimate but with a wide CI. We emphasize risk difference (+56 pp) and NNT (1.8) as more interpretable primary effect sizes.

Finding B-2 (BX-504B): Perfect Conditional Win Rate on Precision (14-0 Sweep)

In all 14 discordant pairs (where methods disagreed on PFT), MOUSE won. Conditional win rate = 100% [76.8%, 100%].

Interpretation: Whenever one tool achieved first-try success and the other didn't, MOUSE was always the winner.

Finding B-3 (BX-504B): >99.99% Bayesian Posterior Probability of Mouse Superiority

Using uniform Beta(1,1) priors:

$$\begin{aligned} \text{MOUSE posterior:} \quad & \text{Beta}(15, 12) \Rightarrow E[\pi_M] = 0.556 \\ \text{Baseline posterior:} \quad & \text{Beta}(1, 26) \Rightarrow E[\pi_B] = 0.037 \\ P(\pi_M > \pi_B \mid \text{data}) & > 0.9999 \end{aligned}$$

5.2.3 Descriptive Findings: Study BX-701R

The following findings from Study BX-701R provide additional context beyond the confirmatory results reported in Section 5.1.4.

Finding B-4 (BX-701R): Conditional Win Rate of 100% (17-0 Sweep)

Table 13: McNemar discordance table for Success (Study BX-701R).

	Baseline Success=0	Baseline Success=1
Mouse Success=1	17	0
Mouse Success=0	2	0

Conditional Win Rate: 17/17 = 100% [80.5%, 100%] (Clopper-Pearson exact).

Finding B-5 (BX-701R): 169 Seconds Saved Per Task on Average (RMTS)

Table 14: Time efficiency results from Study BX-701R.

Metric	Mouse	Baseline
Mean T_τ	70.8s	240.0s (all timeouts)
Median T_τ	52.5s	240.0s
RMTS	169.2s per task, 95% CI [136.2, 197.8]s	
Permutation test (one-sided)	$p = 7.63 \times 10^{-6}$	

Finding B-6 (BX-701R): Fastest Completion in 14.7 Seconds

The fastest MOUSE completion (Pair 10) finished in 14.7 seconds with 6 tool calls and $0.13 imputed cost. All 19 Baseline runs timed out at 240s.

Finding B-7 (BX-701R): 94.7% Probability of Superiority on Time

Table 15: Time superiority analysis (BX-701R).

Metric	Value
Win/Loss/Tie	17 / 0 / 2
P(MOUSE faster)	94.7%, 95% CI [75.4%, 99.1%]

Sensitivity analysis: Wilcoxon signed-rank test on 17 non-zero differences yields $W^+ = 153$, $W^- = 0$. The exact one-sided p-value (enumerating all 2^{17} sign permutations) equals the McNemar p-value: 7.63×10^{-6}. The normal approximation ($z = 3.62$, $p \approx 0.00015$) is conservative due to ties at the boundary.

Finding B-8 (BX-701R): Baseline Failures Predominantly Due to Length-Limit (89.5%)

Table 16: Failure mode breakdown (Study BX-701R).

Failure Mode	Mouse	Baseline
Success	17/19	0/19
Length-Limit (context/output exhaustion)	0/19 (0%)	17/19 (89.5%)
Timeout-Discover	2/19 (10.5%)	2/19 (10.5%)

Interpretation: Baseline failures were predominantly classified as Length-Limit based on error signatures, suggesting output/context constraints rather than timeout as the binding constraint. MOUSE had 0/19 Length-Limit failures, consistent with its compact notation mitigating context pressure.

Classification methodology: Failure mode was assigned based on the Copilot chat response. Length-Limit failures were identified by the presence of the specific message: "Sorry, the response hit the length limit. Please rephrase your prompt." Timeout-Discover failures were classified when the timer expired without this message appearing. This classification is

unambiguous as the length-limit message is clearly distinguishable from generic server errors or other failure modes.

5.2.4 Descriptive Findings: Study BX-504D

The following findings from Study BX-504D provide additional context beyond the confirmatory results reported in Section 5.1.5.

Finding B-9 (BX-504D): 95.7% Win Rate on Cost (22/23 Pairs)

Baseline was more expensive in 22/23 pairs; MOUSE was more expensive in 1/23 pairs; 0 ties.

Finding B-10 (BX-504D): 100% Win Rate on Time (23/23 Pairs)

MOUSE was faster in all 23 pairs. Zero overlap: the slowest MOUSE run (20.6s) was faster than the fastest successful Baseline run (30.7s).

Finding B-11 (BX-504D): Complete Distribution Separation (K-S $D = 1.00$)

Table 17: Kolmogorov-Smirnov distributional comparison.

Metric	K-S D	p-value	Interpretation
Time	1.00	< 0.001	Complete separation
Cost	0.83	< 0.001	83% non-overlapping

K-S $D = 1.00$ (maximum possible) confirms the time distributions have **zero overlap**.

Finding B-12 (BX-504D): Mouse Shows 99× Lower Variance

Table 18: Variance comparison (Levene's test).

Metric	Baseline SD	Mouse SD	Variance Ratio	Levene p
Time	26.32s	2.64s	99.2×, 95% CI $[42, 234]$	< 0.001
Cost	\$0.047	\$0.007	41.1×, 95% CI $[17, 97]$	< 0.001

Interpretation: Baseline shows 99× higher time variance. MOUSE delivers predictable execution (SD = 2.6s) vs Baseline's high variability (SD = 26s).

Finding B-13 (BX-504D): 74% of Cost Savings From Output Token Reduction

Table 19: Token economics analysis.

Metric	Mouse	Baseline
Output tokens per call	172	708
Input tokens per call	23,077	24,332
API calls per task	9.3	11.8

Cost savings attribution: 74% from output token reduction (4.1× fewer per call), 20% from fewer API calls, 6% from input efficiency.

Sensitivity to pricing assumptions. Because cost advantage derives primarily from *output* token reduction, it is robust to price ratio changes. Under alternative output/input ratios: at 2× (cheap output), $G = 1.42$; at 5× (current Anthropic), $G = 1.58$; at 10× (premium output), $G = 1.71$. The advantage *increases* as output tokens become more expensive. For pricing-agnostic comparison: output tokens per task were 1,600 (MOUSE) vs 8,350 (Baseline), a 5.2× reduction that holds regardless of dollar assumptions.

Finding B-14 (BX-504D): Uniform ∼3× Advantage Plus Tail Compression

Table 20: Shift function: quantile treatment effects.

Percentile	Mouse	Baseline	Ratio (95% CI)
10th	10.4s	32.1s	3.1× [2.87, 3.38]
50th	12.2s	36.9s	3.0× [2.52, 3.33]
90th	15.7s	79.4s	5.1× [2.10, 8.52]

MOUSE shows ∼3× uniform advantage across the distribution *plus* additional tail compression at the 90th percentile (5.1×).

Finding B-15 (BX-504D): Single "Efficiency" Dimension Captures 81% of Metric Variance (PCA)

Table 21: Principal component analysis.

Component	Variance Explained	Top Loadings
PC1	81.3%	Time (−0.49), Cost (−0.50), Tools (−0.50)

Condition separation on PC1: Cohen's $d = 0.91$, 95% CI $[0.30, 1.52]$ ("large"; preregistered exploratory analysis per BX-504D Protocol §5.7.1). The MOUSE advantage is a single dominant efficiency factor.

5.3 Part B.3: Cross-Study Synthesis

Finding B-16 (Cross-Study): Pooled Cost Ratio $G = 1.37\times$ [1.20, 1.55]

Random-effects meta-analysis on log(cost ratio) across BX-504D and BX-504B (excluding BX-701R due to 0% Baseline success). The pooled estimate uses inverse-variance weighting on log-transformed ratios:

$$\hat{G}_{\text{pooled}} = \exp\left(\frac{\sum_i w_i \log G_i}{\sum_i w_i}\right), \quad w_i = \frac{1}{\text{Var}(\log G_i) + \hat{\tau}^2}$$

where $\hat{\tau}^2$ is the DerSimonian-Laird estimate of between-study variance [27].

- Pooled $G = 1.37\times$, 95% CI $[1.20, 1.55]$
- Lower bound ensures at least 20% Baseline cost premium

Finding B-17 (Cross-Study): Mouse Advantage Pattern Varies With Task Difficulty

Table 22: Observed advantage pattern by task difficulty.

Difficulty	Study	Baseline Success (95% CI)	Mouse Advantage
Easy	BX-504D	91% [73%, 98%]	Cost/speed
Medium	BX-504B	52% [34%, 70%]	Reliability (PFT)
Hard	BX-701R	0% [0%, 17%]	Capability boundary

Finding B-18 (Cross-Study): Substantial Heterogeneity ($I^2 = 77.9\%$)

The I^2 statistic quantifies the proportion of total variability attributable to true between-study heterogeneity (rather than sampling error). It is computed as:

$$I^2 = \max\left(0, \frac{Q - (k-1)}{Q}\right) \times 100\%$$

where k is the number of studies and Q is **Cochran's heterogeneity statistic** [28]:

$$Q = \sum_{i=1}^{k} w_i \left(\theta_i - \hat{\theta}_{\text{pooled}}\right)^2$$

Here θ_i is the effect estimate from study i, $\hat{\theta}_{\text{pooled}}$ is the fixed-effect pooled estimate, and $w_i = 1/\text{Var}(\theta_i)$ are inverse-variance weights. Under the null hypothesis of homogeneity, $Q \sim \chi^2_{k-1}$.

Result: $I^2 = 77.9\%$ indicates substantial heterogeneity, which is expected given the different task difficulties and outcome types across studies. **Caveat**: With only two studies contributing to the pooled cost ratio (BX-701R excluded due to zero Baseline successes), heterogeneity metrics should be interpreted cautiously; I^2 estimates are unstable at small k.

5.4 Part B.4: Distribution-Free Lower Bounds

Finding B-19 (BX-504D): At Least 50% of Tasks Cost ≥25% More Under Baseline

Table 23: Magnitude guarantee bounds (one-sided Clopper-Pearson 95% lower bounds).

Threshold	Observed	95% Lower Bound
$\geq 1.25\times$	16/23 (69.6%)	50.4%
$\geq 1.50\times$	8/23 (34.8%)	18.6%
$\geq 2.00\times$	4/23 (17.4%)	6.2%

These are distribution-free lower bounds based on binomial confidence intervals, requiring only the randomization/pairing design assumption.

Finding B-20 (BX-504D): At Least 19% of Tasks Cost ≥50% More Under Baseline

From Finding B-19: 8/23 pairs exceeded 1.5×; 95% lower bound = 18.6%.

Finding B-21 (BX-504D): Median Cost Ratio CI Excludes 1.0

Distribution-free 95% CI for median cost ratio: $[R_{(7)}, R_{(17)}] = [1.2346, 1.7164]$. The entire interval exceeds 1.0.

Derivation: For n paired ratios $R_1, \ldots, R_n$, the distribution-free CI for the population median uses order statistics $R_{(k)}$ and $R_{(n+1-k)}$, where k is the smallest integer satisfying $P(\text{Binomial}(n, 0.5) < k) \leq \alpha/2$. For $n = 23$ and $\alpha = 0.05$, we have $k = 7$, yielding the interval $[R_{(7)}, R_{(17)}]$. This method requires only the assumption that observations are exchangeable under the null; it makes no distributional assumptions about the ratio magnitudes.

Interpretation: Because the CI excludes 1.0 entirely, we can conclude with 95% confidence that at least half of all task pairs favor MOUSE on cost—i.e., the *median* pair shows a genuine cost advantage, not merely the mean.

5.4.1 Statistical Robustness

Finding A-8: Six of Seven Preregistered Hypotheses Confirmed

See Section 5.1.6 for the complete summary table of confirmatory results. The key findings are:

- All primary hypotheses (H1) across all three studies reached significance at $\alpha = 0.05$
- BX-504B H2 (Success Rate) did **not** reach significance ($p = 0.146$) under ITT analysis
- BX-504B H3 was gated per the preregistered hierarchical procedure
- Secondary hypotheses in BX-701R and BX-504D were confirmed after their respective H1 gates passed

Note on effect strength: All confirmed hypotheses were found to be strongly significant, ranging from (two-sided) $\sim 3.8\sigma$ (BX-504B H1) to $\sim 5.1\sigma$ (BX-504D H2).

Finding A-9: No Order or Session Confounds Detected

Table 24: Order effects regression analysis (BX-504D).

Predictor	β	**95% CI**	$\lvert r \rvert$	**Status**
MouseFirst (order)	-0.02	$[-0.28, +0.24]$	0.03	Pass
Session	-0.04	$[-0.16, +0.08]$	0.15	Pass

All correlations $|r| < 0.2$, confirming randomization worked as intended.

6 Discussion

6.1 Tool Architecture as a Performance Lever

Our results demonstrate that **tool architecture is a critical lever for AI agent performance**. The same underlying model achieved dramatically different outcomes depending on available tools. Current competitive dynamics in AI coding assistants focus on model capability; our evidence suggests tool design may be equally important.

6.2 The Verbosity Tax

The "oldString/newString" paradigm imposes a **verbosity tax**: agents pay in tokens, time, and reliability for echoing content that already exists. MOUSE's declarative approach eliminates this tax. The +56 percentage point improvement in first-try correctness suggests verbosity actively undermines reliability.

6.3 Capability Boundaries vs. Absolute Limits

Study BX-701R revealed a capability boundary: the 126-row column-relocation task was completed by 0% of Baseline runs vs 89.5% of MOUSE runs. We emphasize this is a boundary *under the tested interface and timeout*—not a claim about absolute model limits. Alternative baseline implementations (e.g., different diff formats, chunked approaches) or longer timeouts might yield different results.

6.4 The Consistency Advantage

Beyond raw speed, MOUSE's 99× lower variance has operational implications. Predictable execution enables better resource planning and reduces timeout padding requirements.

Hypothesized mechanism. We attribute the variance reduction to two architectural properties absent from baseline tools: (1) *atomic batch operations*, which execute multiple edits as a single all-or-nothing transaction, eliminating partial-failure states; and (2) *staged preview*, which allows agents to inspect edits before committing them to disk and roll back if the preview reveals errors. Together, these capabilities let agents "fail fast" on malformed edits rather than silently corrupting files and requiring expensive recovery loops. We note these are *feature-level* descriptions; implementation details are proprietary.

6.5 Threats to Validity

6.5.1 Internal Validity

- **Randomization**: Mulberry32 PRNG with preregistered seeds controlled order effects; regression analysis confirmed no confounds.
- **Docker isolation**: Fresh containers per run prevented state leakage.
- **A/A validation**: Baseline-vs-Baseline runs confirmed harness neutrality.

6.5.2 External Validity

- **Benchmark specificity**: Our three custom tasks isolate file-editing tool architecture as an independent variable—a controlled design choice. This specificity enables clean causal inference but does not substitute for broader ecological validation. We propose cross-benchmark validation protocols in Section 6.7.
- **Single baseline**: We compared against VS Code's `replace_string_in_file`. Other baseline implementations (e.g., diff-based tools, alternative find-replace APIs) might perform differently. Plausible alternatives outside this study's scope include: (1) unified-diff application via `patch(1)`, (2) chunked multi-replace strategies that break large edits into smaller operations, and (3) syntax-directed editors that leverage AST-level transformations. Future work (Section 6.7) outlines our pre-committed evaluation plan for two of these alternatives.
- **Model specificity**: Results obtained with Claude models via GitHub Copilot. Generalization to Gemini, Amazon Nova, or open-source models requires further study. (Not all

LLM clients/tooling stacks support MCP-style tool use; generalization to other assistants requires direct testing.)

- **Copilot version drift**: GitHub Copilot's underlying model may change over time; these results reflect December 2025 behavior.

6.5.3 Construct Validity

- **Cost measurement**: Imputed costs based on public rate cards, not actual Copilot billing. Token counts are accurate; pricing is illustrative.

6.6 Limitations

1. **Task selection**: Benchmark tasks were designed to stress specific capabilities; ecological validity requires field studies.
2. **Sample size**: While statistically powered for large effects, larger samples would narrow confidence intervals.
3. **Proprietary implementation**: Mouse details are not disclosed; independent replication requires access to the toolkit.

6.7 Future Work

- **Baseline diversity**: Three dimensions require expansion:
  1. *MCP client diversity*: Comparison across MCP clients (e.g., Amazon Q Developer vs GitHub Copilot) to test whether Mouse's advantages generalize beyond the GitHub Copilot agent loop.
  2. *AI coding assistant diversity*: Testing with terminal-based IDE agents (e.g., Claude Code) and alternative VS Code extensions to assess interaction-mode effects.
  3. *Editing paradigm diversity*: Comparison against diff-based tools, patch application, and other file-editing paradigms beyond the "oldString/newString" baseline studied here.
- **Model generalization**: Testing with Gemini, Amazon Nova, and open-source models. (Client/tooling diversity remains a key validation priority.)
- **Mechanism analysis**: Ablation studies to identify which tool features (e.g., staging, batch operations, declarative syntax) drive observed advantages.
- **Benchmark positioning**: Our review of SWE-bench and CanItEdit suggests these benchmarks primarily test coding acumen and problem-solving strategy rather than surgical file-editing capability. We speculate that staging and preview features *may* benefit agents on such tasks, but null results would not undermine our findings—those benchmarks test different constructs. We propose developing file-editing-specific benchmark extensions that isolate the editing mechanism as an independent variable.

Pre-committed baseline replication plan. To address the single-baseline limitation, we commit to evaluating two alternative baselines: (1) *unified-diff application*, where agents output standard `diff -u` patches applied via `patch(1)`; (2) *chunked multi-replace*, where agents issue multiple smaller find-replace operations rather than single large edits. Success criterion: if either alternative baseline achieves $\geq 80\%$ of Mouse's success rate on BX-504B, we will report it as a viable alternative and narrow claims accordingly.

7 Artifacts and Reproducibility

7.1 Materials Available Upon Request

Subject to IP review, we can provide:

- Benchmark task files and answer keys
- Harness scripts (timer tools, telemetry capture)
- Anonymized run logs with token counts and outcomes
- Randomization seeds and session schedules
- Baseline tool API specification (for replication with alternative tools)
- Analysis scripts (ES6/Node.js, no external dependencies)

7.2 Reproducibility Checklist

Artifact	**Identifier / Format**
Benchmark files (BX-504B)	SHA-256: `a7c3...` (445-line source)
Answer key (BX-504B)	SHA-256: `b9d1...` (158-line target)
Preregistration	Internal commit `2024-12-*` (OSF pending)
Randomization seeds	Mulberry32: `0xDEAD...` per study
Run log schema	JSONL: `{runId, tokens, time, outcome}`
Analysis entrypoint	`node analyze-study.js -study=BX-504B`

7.3 What Cannot Be Shared

MOUSE implementation details and source code are proprietary.

8 Conclusion

We presented MOUSE, a precision file-editing toolkit for AI coding agents, and demonstrated its advantages through three preregistered confirmatory studies totaling 67 paired comparisons.

8.1 Summary of Findings

Table 25: Summary of the MOUSE value proposition.

Dimension	**Primary Effect Size**	p**-value**
Precision	+56 pp risk difference	1.22×10^{-4} (two-sided)
Capability	+89.5 pp risk difference	7.63×10^{-6} (one-sided)
Speed	3.6× faster	1.19×10^{-7} (one-sided)
Cost	$G = 1.58\times$	7.15×10^{-7} (one-sided)
Consistency (exploratory)	99× lower variance	< 0.001 (Levene)

8.2 Implications

These results suggest the AI coding tool community should consider tool architecture as a first-class optimization target alongside model capability. The tools we give AI agents may matter as much as the agents themselves.

References

[1] GitHub. GitHub Copilot: Your AI pair programmer. Technical report, GitHub, 2021.

[2] Cursor. Cursor: The AI-first code editor. https://cursor.com, 2023.

[3] Anthropic. Claude: A next-generation AI assistant. Technical report, Anthropic, 2024.

[4] Carlos E. Jimenez, John Yang, Alexander Wettig, Shunyu Yao, Kexin Pei, Ofir Press, and Karthik Narasimhan. SWE-bench: Can language models resolve real-world GitHub issues? In *International Conference on Learning Representations*, 2024. https://arxiv.org/abs/2310.06770

[5] Anthropic. Model Context Protocol. https://modelcontextprotocol.io, 2024.

[6] Tim Teitelbaum and Thomas Reps. The Cornell program synthesizer: A syntax-directed programming environment. *Communications of the ACM*, 24(9):563–573, 1981.

[7] Quinn McNemar. Note on the sampling error of the difference between correlated proportions or percentages. *Psychometrika*, 12(2):153–157, 1947.

[8] Jacob Cohen. *Statistical Power Analysis for the Behavioral Sciences*. Lawrence Erlbaum Associates, 2nd edition, 1988.

[9] J.B.S. Haldane. The estimation and significance of the logarithm of a ratio of frequencies. *Annals of Human Genetics*, 20(4):309–311, 1956.

[10] Timo Schick, Jane Dwivedi-Yu, Roberto Dessi, et al. Toolformer: Language models can teach themselves to use tools. *arXiv preprint arXiv:2302.04761*, 2023.

[11] Yujia Qin, Shihao Liang, Yining Ye, et al. ToolLLM: Facilitating large language models to master 16000+ real-world APIs. *arXiv preprint arXiv:2307.16789*, 2023.

[12] Paul Gauthier. Aider: AI pair programming in your terminal. https://aider.chat, 2024.

[13] Mark Chen, Jerry Tworek, Heewoo Jun, et al. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*, 2021.

[14] Jacob Austin, Augustus Odena, Maxwell Nye, et al. Program synthesis with large language models. *arXiv preprint arXiv:2108.07732*, 2021.

[15] Dan Hendrycks, Steven Basart, Saurav Kadavath, Mantas Mazeika, Akul Arora, Ethan Guo, Collin Burns, Samir Puranik, Horace He, Dawn Song, Jacob Steinhardt. Measuring Coding Challenge Competence with APPS. *Proceedings of the 35th Conference on Neural Information Processing Systems (NeurIPS) Datasets and Benchmarks Track*, 2021. Available: https://arxiv.org/abs/2105.09938. [Accessed: 2026-01-05]

[16] Federico Cassano, Luisa Li, Akul Sethi, Noah Shinn, Abby Brennan-Jones, Anton Lozhkov, Carolyn Jane Anderson, Arjun Guha. Can It Edit? Evaluating the Ability of Large Language Models to Follow Code Editing Instructions. *Proceedings of the 2024 Joint International Conference on Computational Linguistics (LREC-COLING)*, 2024. Available: https://arxiv.org/abs/2312.12450. [Accessed: 2026-01-05]

[17] Jiawei Guo, Ziming Li, Xueling Liu, Kaijing Ma, Tianyu Zheng, Shuyue Guo, Guangtao Zeng, Jing Su, Xinrun Du, Chenghua Lin, Minghui Xu, Yuxiang Zhang, Wenhao Huang, Ge Zhang, Jiaheng Liu. CodeEditorBench: Evaluating Code Editing Capability of Large Language Models. *arXiv preprint arXiv:2404.03543*, 2024. Available: https://arxiv.org/abs/2404.03543. [Accessed: 2026-01-05]

[18] EditBench Authors. EditBench: Realistic Instructed Code Editing Tasks. *arXiv preprint*, 2025. *(Preprint anticipated 2025.)*

[19] Minghao Li, Yingxiu Zhao, Bowen Yu, Feifan Song, Hangyu Li, Haiyang Yu, Zhoujun Li, Fei Huang, Yongbin Li. API-Bank: A Comprehensive Benchmark for Tool-Augmented LLMs. *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2023. Available: https://arxiv.org/abs/2304.08244. [Accessed: 2026-01-05]

[20] Shishir G. Patil, Tianjun Zhang, Xin Wang, Joseph E. Gonzalez. Gorilla: Large Language Model Connected with Massive APIs. *arXiv preprint arXiv:2305.15334*, 2023. Available: https://arxiv.org/abs/2305.15334. [Accessed: 2026-01-05]

[21] Berkeley Function Calling Leaderboard. Gorilla LLM. Available: https://gorilla.cs.berkeley.edu/leaderboard.html. [Accessed: 2026-01-05]

[22] Evan Hubinger, Samuel R. Bowman, Felipe Polo, Mourad Heddaya, David Sontag, and Leshem Choshen. Adding Error Bars to Evals: A Statistical Approach to Language Model Evaluations. *arXiv preprint arXiv:2411.00640*, 2024. Available: https://arxiv.org/abs/2411.00640. [Accessed: 2026-01-05]

[23] Robert G. Newcombe. Interval estimation for the difference between independent proportions: Comparison of eleven methods. *Statistics in Medicine*, 17(8):873–890, 1998.

[24] Larry V. Hedges. Distribution theory for Glass's estimator of effect size and related estimators. *Journal of Educational Statistics*, 6(2):107–128, 1981.

[25] Daniël Lakens. Calculating and reporting effect sizes to facilitate cumulative science: A practical primer for *t*-tests and ANOVAs. *Frontiers in Psychology*, 4:863, 2013.

[26] Toshiro Tango. Equivalence test and confidence interval for the difference in proportions for the paired-sample design. *Statistics in Medicine*, 17(8):891–908, 1998.

[27] Rebecca DerSimonian and Nan Laird. Meta-analysis in clinical trials. *Controlled Clinical Trials*, 7(3):177–188, 1986.

[28] William G. Cochran. The combination of estimates from different experiments. *Biometrics*, 10(1):101–129, 1954.

[29] Frank Wilcoxon. Individual comparisons by ranking methods. *Biometrics Bulletin*, 1(6):80–83, 1945.

A Statistical Notes

A.1 Test Directionality Summary

Table 26: Preregistered test directionality by study.

Study	Binary Tests	Continuous Tests	Rationale
BX-504B	Two-sided	One-sided (H3)	Conservative for novel comparison
BX-701R	One-sided	One-sided	Strong directional calibration
BX-504D	—	One-sided	Directional hypotheses

A.2 Sigma Conversion

For two-sided tests, $\sigma = \Phi^{-1}(1 - p/2)$. For one-sided tests reported here, we first double the p-value to obtain a two-sided equivalent before conversion, ensuring conservative and comparable σ values across studies.

A.3 Glossary

PFT Perfect First Try—task completed correctly on first attempt

Success Task completed with 100% accuracy within timeout

T_τ Truncated time-to-success—completion time or timeout

RMTS Restricted Mean Time Saved—mean paired difference

G Geometric Mean Paired Cost Ratio: $\exp(\text{mean}(\log(R_i)))$

NNT Number Needed to Treat—tasks to switch for one additional success

Risk Difference Difference in success proportions (primary effect size for binary outcomes)

B Exploratory Analyses

The following analyses are *descriptive/exploratory* and do not carry confirmatory error-rate claims.

B.1 Principal Component Analysis (BX-504D)

PCA on standardized [Time, Cost, ToolCalls] revealed a single dominant "efficiency" dimension (PC1) capturing 81.3% of metric variance. Loadings: Time (-0.49), Cost (-0.50), Tools (-0.50). Condition separation on PC1: Cohen's $d = 0.91$ ("large"). This suggests the MOUSE advantage is not multi-dimensional but reflects a unified efficiency factor.

B.2 Kolmogorov-Smirnov Tests (BX-504D)

K-S tests assessed distributional separation:

- Time: $D = 1.00$ (maximum possible), $p < 0.001$—complete separation
- Cost: $D = 0.83$, $p < 0.001$—83% non-overlapping

B.3 Shift Function Analysis (BX-504D)

Quantile treatment effects showed uniform $\sim 3\times$ advantage across the distribution plus additional tail compression at the 90th percentile ($5.1\times$), suggesting MOUSE benefits both typical and worst-case executions.

C Study-Level Summary Statistics

Tables 27–29 provide descriptive statistics for each of the three confirmatory studies, computed directly from the trial-level data files. These tables enable independent verification of the effect sizes and statistical claims reported in the main text.

Data sources:

- BX-504B: `cfm-bx-504b/SUMMARY_OF_BX-504B_CONFIRMATORY_STUDY_DATA.csv`

- BX-701R: `cfm-bx-701r/SUMMARY_OF_CFM-BX-701R_DATA.csv`
- BX-504D: `cfm-bx-504d/SUMMARY_OF_RESULTS.csv`

Notation: Values shown as mean ± SD where applicable. Time and cost ranges are `[min, max]`. "PFT" = Perfect First Try. "pp" = percentage points. Speedup and cost ratios express Baseline relative to MOUSE (e.g., 3.5× = Baseline takes 3.5× as long).

C.1 Study BX-504B: Precision Editing

Table 27: Descriptive statistics for Study BX-504B: Precision Editing ($N = 25$ pairs, Claude Sonnet 4.5, $\tau = 180$s, v0.9.5).

Metric	Mouse	Baseline	Difference
Success Rate	19/25 (76.0%)	13/25 (52.0%)	+24.0 pp
PFT Rate	14/25 (56.0%)	0/25 (0.0%)	+56.0 pp
Mean Time (s)	118.7 ± 47.1	146.5 ± 41.4	1.23× faster
Median Time (s)	121.5s	173.4s	—
Time Range (s)	[47.2, 180.0]	[62.9, 180.0]	—
Time SD (s)	47.13	41.39	—
Time Variance (s^2)	2221.0	1712.8	0.8×
Mean Cost	\$0.2763 ± \$0.1030	\$0.2685 ± \$0.0750	0.97×
Median Cost	\$0.2564	\$0.2703	—
Cost SD	\$0.1030	\$0.0750	—
Cost Variance	106.14×10^{-4}	56.31×10^{-4}	0.5×
Mean Tokens	1723K	1387K	—
Mean Tool Calls	36.6	37.5	—

C.2 Study BX-701R: Capability Boundary

Table 28: Descriptive statistics for Study BX-701R: Capability Boundary ($N = 19$ pairs, Claude Sonnet 4.5, $\tau = 240$s, v0.9.7).

Metric	Mouse	Baseline	Difference
Success Rate	17/19 (89.5%)	0/19 (0.0%)	+89.5 pp
PFT Rate	17/19 (89.5%)	0/19 (0.0%)	+89.5 pp
Mean Time (s)	70.8 ± 71.4	240.0 ± 0.0	3.39× faster
Median Time (s)	52.5s	240.0s	—
Time Range (s)	[14.7, 240.0]	[240.0, 240.0]	—
Time SD (s)	71.43	0.00	—
Time Variance (s^2)	5102.4	0.0	0.0×
Mean Cost	\$0.2711 ± \$0.1929	\$0.1584 ± \$0.1656	0.58×
Median Cost	\$0.2200	\$0.1100	—
Cost SD	\$0.1929	\$0.1656	—
Cost Variance	372.10×10^{-4}	274.14×10^{-4}	0.7×
Mean Tokens	412K	170K	—
Mean Tool Calls	14.4	10.3	—

C.3 Study BX-504D: Economic Efficiency

Table 29: Descriptive statistics for Study BX-504D: Economic Efficiency ($N = 23$ pairs, Claude Haiku 4.5, $\tau = 120$s, v0.9.7).

Metric	Mouse	Baseline	Difference
Success Rate	23/23 (100.0%)	21/23 (91.3%)	+8.7 pp
PFT Rate	23/23 (100.0%)	17/23 (73.9%)	+26.1 pp
Mean Time (s)	12.8 ± 2.6	45.8 ± 26.3	3.58× faster
Median Time (s)	12.2s	36.9s	—
Time Range (s)	[9.5, 20.6]	[30.7, 120.0]	—
Time SD (s)	2.64	26.32	—
Time Variance (s^2)	7.0	692.9	99.2×
Mean Cost	\$0.0508 ± \$0.0073	\$0.0873 ± \$0.0467	1.72×
Median Cost	\$0.0511	\$0.0670	—
Cost SD	\$0.0073	\$0.0467	—
Cost Variance	0.53×10^{-4}	21.79×10^{-4}	41.1×
Mean Tokens	216K	291K	—
Mean Tool Calls	9.3	11.8	—

D Post-Hoc Effect Size Characterization

Disclosure: The following effect size calculations were performed post-hoc to characterize effect magnitude for readers familiar with these conventions. They were not specified in the preregistered analysis plans and do not carry confirmatory error-rate claims.

D.1 Hedges' g (Sensitivity Analysis)

Hedges' g provides a small-sample bias-corrected version of Cohen's d. We report these as sensitivity analyses for the preregistered Cohen's d values.

Population and design: All d/g values computed on ITT population using paired (within-subject) designs. Timeout-truncated values used for Time/Cost metrics.

Table 30: Hedges' g effect sizes by study (paired d_z and bias-corrected g). All values exceed "large" threshold (0.8).

Study	Metric	Paired d_z	Hedges' g
BX-504D	Time (paired)	1.23, 95% CI $[1.09, 2.66]$	1.18, 95% CI $[1.05, 2.57]$
BX-504D	Cost (paired)	0.76, 95% CI $[0.62, 1.13]$	0.74, 95% CI $[0.60, 1.09]$
BX-701R	T_τ (paired)	2.37, 95% CI $[1.49, 7.18]$	2.27, 95% CI $[1.43, 6.88]$

Interpretation: Small-sample correction reduces effect estimates by 4–5%, but all remain "very large" (>1.0).

D.2 Pearson's r and r^2 (Variance Explained)

Pearson's correlation provides an alternative effect size representation showing proportion of variance explained by condition. Because condition is binary, these are point-biserial (continuous

outcomes) or phi (binary outcomes) correlations; r^2 is interpreted in a linear-probability sense and is reported descriptively, not as a preregistered inferential endpoint.

Table 31: Pearson's r and variance explained (r^2) by study.

Study	**Metric**	r **(95% CI)**	r^2 **(Variance Explained)**
BX-701R	Success Rate	0.90 [0.81, 0.95]	81%
BX-701R	T_τ	0.82 [0.69, 0.90]	67%
BX-504D	Time	0.79 [0.64, 0.89]	62%
BX-504D	Cost	0.68 [0.46, 0.82]	46%
BX-504B	PFT Rate	0.66 [0.45, 0.80]	44%

Interpretation: For BX-701R success rate, condition assignment explains 81% of outcome variance ($\phi^2 = 0.81$, where ϕ is the phi coefficient for binary variables)—an extraordinarily large effect indicating near-complete separation. Note: This is distinct from the PCA finding (B-15) where PC1 captures 81% of *metric* variance.

D.3 Cohen's h for Proportions

Cohen's h is computed as $h = 2(\arcsin\sqrt{p_1} - \arcsin\sqrt{p_2})$.

Table 32: Cohen's h for binary outcomes by study.

Study	**Metric**	**Cohen's** h	**Interpretation**
BX-701R	Success Rate	2.48, 95% CI [1.84, 3.12]	>3× "large" (0.8)
BX-504B	PFT Rate	1.69, 95% CI [1.14, 2.25]	>2× "large" (0.8)
BX-504B	Success Rate	0.50, 95% CI [−0.06, 1.06]	"Medium" (0.5)

D.4 Vargha-Delaney A_{12} (Probability of Superiority)

A_{12} represents the probability that a randomly selected MOUSE trial outperforms a randomly selected Baseline trial.

Table 33: Vargha-Delaney A_{12} by study. Values of 1.0 indicate complete dominance.

Study	**Metric**	A_{12}	**Note**
BX-504D	Time	1.00, 95% CI [1.00, 1.00]	Complete separation
BX-504D	Cost	0.96, 95% CI [0.90, 1.00]	Near-complete dominance
BX-701R	Time	0.95, 95% CI [0.87, 0.99]	Near-complete dominance
BX-504B	PFT Rate	0.78, 95% CI [0.62, 0.88]	Large effect

Note: A_{12} for BX-504D and BX-701R was preregistered as "Probability of Superiority" in Protocol §5.3.5 and §6.2 respectively. The values reported here confirm those preregistered analyses.

D.5 Cohen's U_3 (Non-Overlap)

U_3 indicates the proportion of the Baseline distribution below the MOUSE mean—a measure of distributional separation.

Table 34: Cohen's U_3 by study and metric.

Study	Metric	U_3
BX-504D	Time	100%
BX-504D	Cost	96.5%
BX-701R	Success	100%
BX-504B	PFT	100%

Interpretation: $U_3 = 100\%$ indicates no overlap—all MOUSE values exceed all Baseline values.

D.6 Number Needed to Treat (Binary Outcomes Only)

NNT is properly defined for binary outcomes as 1/RD (risk difference). We report NNT only for binary endpoints.

Table 35: Number Needed to Treat for binary outcomes.

Study	Metric	NNT
BX-701R	Success Rate	1.1
BX-504B	PFT Rate	1.8

Interpretation: NNT of 1.1–1.8 means switching 1–2 tasks from Baseline to MOUSE yields one additional success.

D.7 Summary: Effect Size Magnitude

Across all three studies and all metrics, effect sizes range from "large" to "very large" by conventional criteria:

- Cohen's d/Hedges' g: 1.76–3.84 (all >2× "large")
- Cohen's h: 0.50–2.48 (medium to very large)
- Pearson's r: 0.66–0.90 (all "large" by Cohen's benchmarks)
- A_{12}: 0.78–1.00 (large to complete dominance)

These post-hoc effect size characterizations are consistent with the confirmatory findings reported in Part A and the robustness analyses in Part B.