FITBENCH: BENCHMARKING HUMAN–AI COLLABORATION ACROSS OPTION-PICKING-DRIVEN DECISION STYLES

Anonymous authors
Paper under double-blind review

ABSTRACT

Existing benchmarks primarily measure Large Language Models (LLM) or incorporate Harness's task capabilities and performance, which can accurately evaluate the agent's ability and efficiency in reasoning and thinking tasks. However, humans in reality have different decision-making personalities and preferences. Using a single evaluation criterion to assess AI cannot truly reflect the value of this performance in real-world applications, nor can it be used to measure Human–AI collaboration efficiency. It is jointly determined by the task itself, AI performance, and human decision-making personality. Therefore, if it is to be evaluated, improvements need to be made to existing benchmarks by incorporating human decision-making personality variables. To this end, we propose FitBench, which represents human side differences as six structured human slots under the same task conditions, and systematically evaluates Human–AI collaboration performance under different decision-making personalities, while comparing the impact of different LLM and Harness configurations. FitBench models the collaboration process as three interconnected subsystems and introduces an optional Interactive Structured Assertion (iSA) interaction adapter to enhance the observability of slot differences. In the current experimental results, the range and variance of success rates across the six slots for the same actual task show significant differences; both increase in the recorded iSA control. In addition, we can select the optimal Harness configuration for each human decision-making personality slot while keeping the LLM layer fixed, thereby significantly improving the efficiency of AI collaboration with humans of different decision-making personalities. Thus, FitBench provides a unified evaluation framework for separating model performance from human decision-making differences, analyzing the conditional interaction between human slots and AI execution configuration, and further studying collaboration configurations tailored to different decision-making personalities.

1 INTRODUCTION

A benchmark provides a reproducible basis for comparing systems under fixed tasks, data, protocols, and metrics. It turns an open-ended question about capability into a reviewable set of tasks, scoring rules, and reporting protocols. Large-scale resources such as ImageNet established a common data foundation for visual evaluation; subsequent protocols expanded to the multitask and diagnostic suites of GLUE and SuperGLUE and the multi-scenario, multi-metric, transparently documented framework of HELM (Deng et al., 2009; Wang et al., 2019b;a; Liang et al., 2023). These studies show that a benchmark does more than produce a score: it specifies what is evaluated, controls the conditions of comparison, reveals capability boundaries, and supports extension.

Benchmarks for language models and agents have made knowledge, reasoning, planning, tool use, and task success measurable. MMLU and BIG-bench primarily evaluate models using fixed inputs and predefined criteria; MT-Bench and Chatbot Arena extend evaluation to conversational quality; and AgentBench and TravelPlanner measure execution in multistep environments (Hendrycks et al., 2021; Srivastava et al., 2022; Zheng et al., 2023; Liu et al., 2024; Xie et al., 2024). These benchmarks primarily ask whether an AI system can complete a task. Human input and feedback are usually

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.

fixed or abstracted away, so distinguishing collaboration efficiency across human decision behaviors is not their primary objective.

Model capability alone does not determine the efficiency of collaboration. The same high-performing model may behave differently under different solver loops, tools, memory mechanisms, and interaction protocols; it may also impose different decision burdens on people who commit quickly, explore alternatives, rely on advice, or repeatedly revise goals. Human decision behavior does not change the task world's transition rules, but it can affect candidate selection, feedback density, goal updates, and the number of interaction turns. Collaboration efficiency therefore encompasses task outcomes, interaction cost, and convergence. The strongest model in isolation need not be the best collaborator under every human behavior condition.

Research on personalization, user simulation, and behavior-conditioned evaluation has established the importance of user differences. LaMP conditions prediction and generation of user history, Non-Collaborative User Simulators measure how distinct non-cooperative behaviors affect tool agents, and Collaborative Gym evaluates both outcomes and processes in human–agent collaboration (Salemi et al., 2024; Shim et al., 2026; Shao et al., 2026a). Together, these efforts motivate a controlled benchmark that explicitly specifies the selection, feedback and goal-revision rules and records their effects on outcomes, cost, and convergence. FitBench addresses this need by treating structured human behavior as an independent evaluation factor.

FitBench measures how agents perform with different decision behaviors under identical task conditions and whether relative adaptation is reflected in collaboration efficiency. Its central questions are whether the six human slots, structured from the literature and questionnaire to characterize decision-making styles, produce interpretable outcome differences and whether an execution configuration tailored to each slot can outperform a uniform one. Adaptation cannot be assessed from a single success rate; it must also account for outcome quality, interaction cost, and stability. We therefore define task $\times$ human slot $\times$ LLM $\times$ Harness as the basic evaluation space and use the persona gap as a diagnostic measure of differences, rather than as evidence of a globally optimal match.

FitBench draws on the decision-psychology literature and a questionnaire study involving hundreds of respondents to define six executable human slots: `decisive`, `exploratory`, `deferential`, `shifting`, `contradictory`, and `low_engagement`. It models interaction as a state–wish–trace loop and introduces an optional interaction adapter, `iSA`, to make slot differences more observable. We first hold the Harness fixed while comparing LLMs and slots, then hold the selected LLM fixed while comparing Harnesses and slots. Our contributions are:

- We introduce FitBench, a benchmark that treats structured human decision behavior as a controlled variable alongside model and Harness capabilities.
- We define six executable human slots with rules that can be reproduced in an interaction loop and evaluated using task outcomes, process traces, and interaction cost.
- We provide a closed-loop decomposition and an optional interaction adapter that make differences in human behavior auditable without treating slots as clinical personality categories.

2 Related Work

AI evaluation has progressed from static capability measurement to execution in interactive environments. MMLU measures broad knowledge and understanding, BIG-bench collects diverse tasks, and MT-Bench and Chatbot Arena compare multi-turn responses and human preferences (Hendrycks et al., 2021; Srivastava et al., 2022; Zheng et al., 2023). AgentBench evaluates reasoning and decision-making in several environments, while TravelPlanner tests planning under realistic constraints (Liu et al., 2024; Xie et al., 2024). SWE-bench evaluates software repair using real repositories and tests, WebArena provides reproducible web tasks, and GAIA emphasizes retrieval, multimodal understanding, and tool composition (Jimenez et al., 2024; Zhou et al., 2024; Mialon et al., 2024). ReAct alternates reasoning and acting, while Reflexion uses verbal feedback and memory to improve subsequent attempts (Yao et al., 2023; Shinn et al., 2023). AndroidWorld, Terminal-Bench, MCP-Bench, and Gaia2 extend evaluation to parameterized mobile tasks, long-horizon terminal workflows, coordination across tools, and asynchronous environments (Rawles

et al., 2025; Merrill et al., 2026; Wang et al., 2026b; Froger et al., 2026). Benchmarks that explicitly incorporate human collaboration have also advanced: PARTNR studies coordination in embodied tasks, Collaborative Gym supports bidirectional communication and evaluates both simulated and real users, CentaurEval measures the value of human intervention in coding, CollabSkill decomposes skills in real-world matching tasks, DeskCraft defines collaboration protocols for professional desktop workflows, and the Two-Player Alternate Uses Test compares Human–AI and human–human co-creation under controlled conditions (Chang et al., 2025; Shao et al., 2026a; Luo et al., 2026; Shao et al., 2026b; Wang et al., 2026a; Hemmatian et al., 2026). Compared with autonomous agent evaluation, Human–AI collaboration remains less extensively studied, and task types, dimensions of human difference, and reporting protocols are not yet fully standardized. FitBench therefore addresses a narrower complementary question: under fixed cases and LLM conditions, how do explicit rules for selection, feedback, and goal revision interact with Harness organization and affect cost and convergence?

Research on personality, user simulation, and personalization offers another approach to modeling user differences. Generative Agents uses memory, planning, and reflection to simulate persistent behavior; User Simulation with Large Language Models constructs users for task-oriented dialogue; and LaMP evaluates personalization conditioned on user history (Park et al., 2023; Davidson et al., 2023; Salemi et al., 2024). PersonaLens evaluates personalization using user profiles and interaction histories, while a recent survey organizes conversational user simulation by user granularity, simulation target, method, and evaluation dimension (Zhao et al., 2025b; Ni et al., 2026). PersonalLLM models heterogeneous preferences using diverse reward models, SeCom studies memory construction and retrieval, and PrefEval evaluates preference recognition and adherence in long contexts (Zollo et al., 2025; Pan et al., 2025; Zhao et al., 2025a). RGMem separates short-term evidence from slowly changing user traits; Persona-Guided LLM Agents compare explicit and inferred personas in task-oriented dialogue; and P-GenRM models personalized rewards using test-time user scaling (Tian et al., 2026; Shoaeinaeini et al., 2026; Zhang et al., 2026). Personality Alignment, PAL, and FSPO study alignment or reward adaptation for different users (Zhu et al., 2025; Chen et al., 2025; Singh et al., 2026). Non-Collaborative User Simulators distinguish unavailable requests, topic shifts, impatience, and incomplete expression, while Flipping the Dialogue trains user language models for more realistic multi-turn behavior (Shim et al., 2026; Naous et al., 2026). PrefDisco, PAMDP, and Personality Subnetworks examine proactive preference discovery, latent persona alignment, and parameter-level personality structure (Li et al., 2026; Yang et al., 2026; Ye et al., 2026). These studies use user differences for simulation, preference discovery, and adaptation, but their primary aims range from preference consistency and response quality to simulation realism and robustness. They provide a foundation for studying behavior, yet do not center a common matrix of tasks and Harnesses in which acceptance, exploration, goal revision, and feedback investment are compared through outcomes, cost, and convergence. FitBench makes these behavioral rules configurable and auditable and evaluates them alongside task and process measures.

3 METHOD

FitBench is a closed-loop benchmark for comparing Human–AI collaboration across decision behaviors and execution configurations. It separates three sources of variation that are often conflated: the task world determines which transitions are possible; the LLM, supported by an AI execution Harness, generates and organizes candidate actions; and a structured human slot makes selections, provides feedback, and revises goals during interaction. Each slot is a controlled behavior condition rather than a diagnostic category and does not alter task-world dynamics.

3.1 PROBLEM DEFINITION

Each evaluation condition comprises a task, a structured human slot, an LLM, and an AI execution configuration. We use Harness to denote the execution layer surrounding LLM outputs, including the solver, tools, state and memory, interaction protocol, validation, and observability; the LLM/provider remains an independent model variable. The benchmark space is

$$\begin{aligned} \text{FitBench} &= \text{Task} \times \text{HumanSlot} \times \text{LLM} \times \text{Harness}, \\ \text{Harness} &= \text{Solver} \times \text{Tools} \times \text{State/Memory} \times \text{Protocol} \times \text{Validation}. \end{aligned} \tag{1}$$

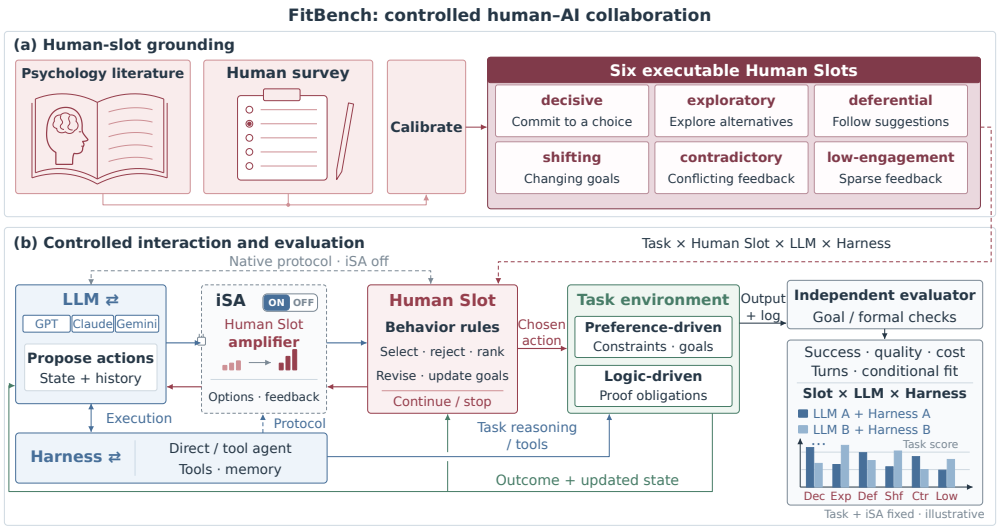

Figure 1: Overview of FitBench. (a) Six executable human slots—decisive, exploratory, deferential, shifting, contradictory, and low_engagement—are calibrated using behavioral literature and a survey. (b) In the controlled interaction loop, an LLM proposes actions; a Harness provides tools and memory; an optional iSA adapter amplifies options and feedback conditioned on the human slot; the human slot selects, rejects, ranks, revises goals, or stops; the task environment evaluates preference-driven constraints or logic-driven proof obligations; and an independent evaluator reports success, quality, cost, turns, and conditional fit across Slot × LLM × Harness conditions.

For case c, a run begins with initial state s_0 and wish w_0. At step t, the LLM proposes a candidate set M_t, slot h selects m_t, and the task world returns outcome o_t and next state s_{t+1}. The observation then updates both the wish and the interaction trace H_t. Controlled comparisons vary only the factor of interest; cases, prompts, seeds, termination rules, and all other components remain fixed. We report task outcomes, process traces, cost, and stability diagnostics. Adaptation is a relational outcome observed across conditions, rather than an isolated LLM score or an unconstrained global optimum.

3.2 Human Slot Design and Calibration

The distinction between continued search and stopping once an acceptable result has been found provides a theoretical reference for structured behavior. Schwartz et al. examined maximizing and satisficing tendencies and their relationships with choice experience and well-being (Schwartz et al., 2002). We use the distinction between searching and stopping as a design reference; the six slots themselves are defined by our operational rules, questionnaire items, and calibration procedure rather than derived directly from that theory.

Human slots are behavioral abstractions designed for benchmarking. First, psychological research on decision styles, preference expression, and Human–AI interaction informed the selection of dimensions directly relevant to interaction. Second, a questionnaire study involving hundreds of respondents collected self-reports and collaboration-related feedback; these responses informed the content and boundaries of the slots. This process supports their use as reproducible behavior conditions, but does not establish clinical categories or a complete behavioral taxonomy.

The six slots are characterized as follows: `decisive` commits quickly and maintains a chosen direction; `exploratory` requests alternatives and information before committing; `deferential` accepts advice or changes direction in response to it; `shifting` changes priorities during interaction; `contradictory` provides partially inconsistent feedback; and `low_engagement` provides sparse feedback and weak preference signals. Each rule affects candidate selection, feedback density, wish updates, or stopping behavior.

Table 1: Structured human slots in FitBench. Slots are controlled behavioral abstractions rather than personality measurements.

Human slot	Operational behavior in the interaction loop
`decisive`	Selects a candidate quickly and commits to a direction, tending to shorten the interaction.
`exploratory`	Requests alternatives and additional information, exploring candidates before committing.
`deferential`	Reveals fewer preferences and accepts advice earlier, remaining sensitive to externally suggested changes of direction.
`shifting`	Changes priorities during interaction, so the desired outcome may shift across turns.
`contradictory`	Provides partially inconsistent feedback, creating tension between stated preferences and accepted outcomes.
`low_engagement`	Provides sparse feedback and weak preference signals, making convergence more dependent on the LLM-side solver.

For candidate $m_t^{(k)} \in M_t$, a slot is represented by the following persona-conditioned utility:

$$U_\phi(m_t^{(k)}, s_t, w_t, H_t, p) = \alpha_p A_t^{(k)} + \beta_p B_t^{(k)} - \gamma_p C_t^{(k)} - \delta_p R_t^{(k)} - \zeta_p Q_t^{(k)}. \tag{2}$$

Here p is the slot descriptor, and $\phi(p) = (\alpha_p, \beta_p, \gamma_p, \delta_p, \zeta_p)$ specifies its weights. A, B, C, R, and Q denote estimated progress, alignment with the current wish, execution cost, risk, and attention or cognitive burden, respectively. The selected move is

$$m_t = \arg\max_{m \in M_t} U_\phi(m, s_t, w_t, H_t, p), \tag{3}$$

or a softmax sample with slot temperature τ_p. These coefficients summarize behavioral priorities rather than parameters of a separately trained psychometric model. The slot changes which available action is selected and therefore changes the trajectory, while leaving the task world unchanged.

3.3 Harness

The Harness is the AI execution layer surrounding LLM outputs. It provides persistent execution, tool calls, state maintenance, validation, and traceability. We group its components into the solver or agent loop, tools and environment interface, state and memory, interaction protocol, guardrails and validation, and observability. The experiments compare Direct, ReAct, Reflexion, Plan-and-Execute, and Self-Refine while keeping the task interface, validation, budget, and trace requirements fixed.

3.4 Three-Subsystem Collaboration Model

The collaboration loop has three connected subsystems: task-world dynamics, LLM candidate generation, and persona-conditioned decision convergence. At each step, the LLM reads the current state, wish, task context, and structured trace through the Harness, produces a finite candidate set, and the human slot selects one candidate. The task world executes the move and writes the outcome back as feedback for the next step. The loop produces move, state, and wish trajectories over a horizon T:

$$M_t = \pi_\theta(s_t, w_t, H_t, c), \qquad m_t = \mathrm{Select}_\phi(M_t, s_t, w_t, H_t), \tag{4}$$

$$(s_{t+1}, o_t) \sim \mathcal{T}(s_t, m_t), \qquad w_{t+1} = \Psi_\eta(w_t, o_t, s_{t+1}, H_t). \tag{5}$$

Here π_θ is the candidate generator, Select_ϕ is the selection mechanism conditioned on the human slot, $\mathcal{T}$ is the task transition kernel, and Ψ_η is the wish update function. In prompt-based implementations, these parameterizations need not correspond to trained modules.

The division of labor among the three subsystems is central. A task-related distance $d_{SW} : \mathcal{S} \times \mathcal{W} \to \mathbb{R}_{\geq 0}$ can be used to diagnose progress:

$$\Delta_t^{\mathrm{prog}} = d_{SW}(s_t, w_t) - d_{SW}(s_{t+1}, w_{t+1}), \qquad v_T = \frac{d_{SW}(s_0, w_0) - d_{SW}(s_T, w_T)}{T}. \tag{6}$$

Positive Δ_t^{prog} indicates progress toward the updated wish, and v_T denotes the average convergence speed. These quantities are diagnostics rather than a universal reward; some tasks do not provide a computable distance. The transition kernel determines what counts as valid progress, such as a feasible itinerary or an accepted proof. In a deterministic setting, $(s_{t+1}, o_t) = (\mathcal{T}_s(s_t, m_t), \mathcal{T}_o(s_t, m_t))$.

The LLM generates candidate actions, while the solver, provider, tools, and protocol determine how candidates are generated, revised, and organized:

$$M_t = \{m_t^{(1)}, \ldots, m_t^{(K)}\}, \qquad M_t \sim \pi_\theta(\cdot \mid s_t, w_t, H_t, c), \tag{7}$$

$$\rho_t^{(k)} = \mathbb{E}_{\hat{p}_\theta(o|s_t, w_t, m_t^{(k)}, H_t, c)}[R_{\text{prog}}(s_t, w_t, m_t^{(k)}, o)] - \lambda_u u_t^{(k)} - \lambda_c \kappa_t^{(k)}. \tag{8}$$

This expression provides a general description of candidate generation. The experiments record candidates, selections, feedback, and state updates; they do not treat uncertainty, cost, or progressibility as separately trained modules. A slot changes the distribution of selected actions, but not $\mathcal{T}$.

3.5 iSA Design

Interactive Structured Assertion (iSA) is a pluggable interaction adapter rather than a solver family. The solver determines how the Harness reasons, calls tools, or revises a plan; the adapter makes it easier to record how each slot selects candidates, provides feedback, and revises goals. It exposes a common set of candidate options and supports acceptance, rejection, ranking, additional constraints, or requests for a new direction. The resulting traces make slot rules observable. In controlled comparisons, preference revelation, wish updates, and process differences measure observability, whereas task outcomes and interaction turns measure efficiency. An increase in persona gap, range, or variance indicates a more dispersed outcome distribution and does not by itself imply better collaboration or preference fit.

The feedback loop uses the dynamic-wish form

$$w_{t+1} = \Psi_\eta(w_t, o_t, s_{t+1}, H_t, p) = w_t + \Delta_t^{\text{fb}} + \Delta_t^{\text{pref}} + \Delta_t^{\text{clar}}. \tag{9}$$

These terms denote the direct effect of the observation, the preference adjustment associated with the slot, and the clarification of an initially ambiguous wish, respectively. This decomposition explains why the adapter can increase observability: a structured result and feedback update w_{t+1} and consequently affect the next candidate set M_{t+1}.

3.6 Evaluation Metrics

Task outcome. Each attempted trajectory receives a graded score $q_i \in [0, 1]$ from a fixed external LLM judge that is blinded to the model, Harness, slot, and prior success label. The judge sees the task, final artifact, and feedback available before final submission. For TravelPlanner, H_i averages scores for hard constraints (met: 1; violated: 0; unverified: 0.5), while E_i measures plan executability and A_i measures alignment with expressed preferences and feedback. We set $q_i = 0.65H_i + 0.15E_i + 0.20A_i$, with an upper bound of 0.49 if a hard constraint is violated and 0.75 if one is unverified. For miniF2F, $L_i = 1$ only when the saved final proof passes Lean without `sorry` or `admit`; A_i measures responsiveness to prior feedback and checker errors. We set $q_i = 0.85L_i + 0.15A_i$, with $A_i = 0.5$ when no prior feedback is observable. Trajectories without an evaluable final artifact, including serving failures and timeouts, receive $q_i = 0$. The graded success rate is

$$\text{SR} = \frac{1}{N} \sum_{i=1}^{N} q_i. \tag{10}$$

Within-case range and variance. For a fixed LLM, Harness, protocol, and budget, let $\text{SR}_{c,h}$ denote the slot success rate for case c and $\overline{\text{SR}}_c = \frac{1}{6} \sum_{h \in \mathcal{H}} \text{SR}_{c,h}$. We average the within-case range and population variance across cases:

$$\begin{aligned} \Delta_{\text{case}} &= \frac{1}{|\mathcal{C}|} \sum_{c \in \mathcal{C}} \left(\max_{h \in \mathcal{H}} \text{SR}_{c,h} - \min_{h \in \mathcal{H}} \text{SR}_{c,h} \right), \\ V_{\text{case}} &= \frac{1}{|\mathcal{C}|} \sum_{c \in \mathcal{C}} \left[\frac{1}{6} \sum_{h \in \mathcal{H}} (\text{SR}_{c,h} - \overline{\text{SR}}_c)^2 \right]. \end{aligned} \tag{11}$$

The denominator is 6 because the six formal slots constitute the population of interest. We compute the range and variance within each case before averaging across cases; we do not compute the variance of six case-aggregated rates. Both quantities describe outcome differences among slots and make variation among slots visible.

Macro success. To avoid directly pooling task units of unequal size, we report the equally weighted mean of the two task-level success rates:

$$\text{Macro SR} = \frac{1}{2}(\text{SR}_{\text{TravelPlanner}} + \text{SR}_{\text{miniF2F}}). \tag{12}$$

Interaction cost and process signals. For trajectory i, let T_i denote the number of turns and $\overline{T} = N^{-1}\sum_i T_i$ their mean. TravelPlanner process signals are the mean numbers of revealed preferences r_i and accepted directions a_i:

$$\overline{R} = \frac{1}{N}\sum_i r_i, \qquad \overline{A} = \frac{1}{N}\sum_i a_i. \tag{13}$$

These preference signals are not defined for miniF2F and are therefore reported as "–" rather than zero.

iSA difference amplification and comparison by slot. Let the original and full-adapter gaps be Δ_{orig} and Δ_{iSA}, respectively, so that $\text{AiSA} = \Delta_{\text{iSA}}/\Delta_{\text{orig}}$. A zero baseline is reported as N/A rather than assigned an artificial multiplier. In the Harness matrix, development-set slot rankings determine the choice for each slot, whereas the global-best strategy uses a single Harness for all slots. The paired gain and per-slot regret are

$$\text{Gain}_{\text{matched}} = \text{SR}_{\text{matched}} - \text{SR}_{\text{global-best}}, \qquad \text{Regret}_h = \text{SR}_{h,\text{best}} - \text{SR}_{h,\text{global-best}}. \tag{14}$$

Range and variance characterize dispersion and do not enter the matching-gain calculation.

For each run, we record the case, human slot, LLM/provider, Harness, protocol, seed, maximum number of turns, candidate set, selection, outcome, wish update, termination state, actual number of turns, revealed preferences, accepted directions, and Lean acceptance. Fixed denominators retain serving failures, timeouts, invalid outputs, and checker errors as failures. We report outcomes, dispersion, turns, and the process signals defined for each task.

4 Experiments

4.1 Goals and Setup

The experiments address three questions: whether the six human slots yield distinguishable outcomes under identical task conditions; how LLM task capability and stability across slots vary under a fixed Harness; and how different Harness organizations affect each slot's outcomes under a fixed LLM. The main experimental matrix is Task × Human Slot × LLM/Harness. We use two primary control schemes. In the LLM comparison, ReAct, iSA, task instances, budgets, and validation are fixed while nine LLMs vary. In the Harness comparison, Qwen-3.7-Max, iSA, task instances, budgets, and validation are fixed while five Harnesses vary. The merged comparison tables use 18 TravelPlanner and 8 miniF2F cases per row; the separate iSA ablation uses the frozen 9-plus-4 case set. Each condition has one fixed replicate, and confidence intervals use cases as the clustering unit. This controlled design separates model capability, Harness organization, and human slots, allowing outcome differences among slots to be attributed to specified experimental conditions.

4.2 Dataset and Task Suites

To cover both subjective negotiation and objective verification, we selected the open-source TravelPlanner and miniF2F suites. TravelPlanner requires itinerary construction under multiple constraints (budget, time, location, transportation, preferences), allowing us to observe preference revelation, plan revision, constraint satisfaction, and final acceptance. It is thus the primary task for evaluating Human-AI negotiation and adaptation. Success is determined by explicit hard constraints and final-plan checks, not LLM self-assessment.

Table 2: LLM comparison under a fixed Harness. Δ_{case} and V_{case} are the mean within-case range and population variance of the six slot success rates.

LLM	Decisive	Exploratory	Deferential	Shifting	Contradictory	Low-engagement	SR	Δ_{case}	V_{case}
GPT-5.4	0.830	0.804	0.836	0.879	0.778	0.824	0.825	0.353	0.03813
GPT-5.5	0.687	0.739	0.676	0.806	0.622	0.755	0.714	0.542	0.06928
GPT-5.6-luna	0.569	0.484	0.534	0.597	0.630	0.683	0.583	0.682	0.10053
Gemini-3.5-flash	0.700	0.686	0.634	0.633	0.660	0.723	0.672	0.579	0.07438
Gemini-3.6-flash	0.634	0.452	0.648	0.409	0.397	0.682	0.537	0.700	0.11533
Gemini-3.7-flash	0.773	0.868	0.773	0.809	0.737	0.859	0.803	0.289	0.04110
Qwen-3.7-Max	0.809	0.755	0.765	0.766	0.778	0.772	0.774	0.203	0.01898
Kimi-K2.6	0.638	0.683	0.786	0.476	0.555	0.739	0.646	0.689	0.11595
DeepSeek-V4-flash	0.698	0.653	0.761	0.623	0.523	0.659	0.653	0.718	0.09663

Table 3: Human Slot $\times$ Harness task success under a fixed LLM.

Harness	Decisive	Exploratory	Deferential	Shifting	Contradictory	Low-engagement	SR	Δ_{case}	V_{case}
Direct	0.791	0.681	0.777	0.741	0.693	0.731	0.736	0.351	0.04765
ReAct	0.809	0.755	0.765	0.766	0.778	0.772	0.774	0.203	0.01898
Reflexion	0.682	0.740	0.743	0.720	0.597	0.725	0.701	0.282	0.03358
Plan-and-Execute	0.651	0.602	0.617	0.665	0.656	0.653	0.641	0.266	0.03771
Self-Refine	0.666	0.573	0.662	0.516	0.405	0.647	0.578	0.413	0.06173

miniF2F provides mathematical, logical, and formal proof tasks. The Harness must generate or repair Lean proofs, and Lean kernel acceptance is the sole correctness criterion. Human slots can influence candidate selection, error-feedback responses, and repair paths, but cannot alter the definition of correctness. This task therefore mainly provides evidence on objective correctness, tool verification, and process robustness; alone, it does not support conclusions about user-preference matching.

Rather than directly adopting the original open-source instances, we augmented the tasks with additional hard constraints to increase difficulty and created similar variants based on the original content, doubling the number of cases. The resulting suite contains 18 TravelPlanner cases and 8 miniF2F cases in total.

4.3 LLM Comparison under a Fixed Harness

We fix the Harness to ReAct and hold the task interface, state–wish–trace representation, interaction adapter, validation, and budget constant, varying only the LLM/provider. Each model is evaluated on the same TravelPlanner and miniF2F units. We report success, within-case range and variance, and macro success.

Qwen-3.7-Max offers the most balanced overall performance while maintaining low dispersion among slots, so we use it as the fixed LLM in the subsequent comparison. Other models differ substantially in task capability, stability across tasks, and dispersion among slots, showing that Fit-Bench captures sensitivity to human behavior conditions in addition to overall model performance. Within model families, newer versions show an overall tendency toward greater variance among slots. This tendency appears in both the GPT and Gemini series: it is more consistent for GPT, whereas Gemini shows a local decline. It should therefore be interpreted as a non-monotonic pattern in the current table, not as a general rule or a causal effect of model upgrades. Reporting slot variance alongside aggregate success helps expose these differences in human behavior.

4.4 Harness Comparison under a Fixed LLM

Based on the preceding results, we select Qwen-3.7-Max as the fixed LLM because it achieves relatively high success rates with the smallest dispersion among slots. Holding the model constant allows us to compare slot outcomes across Harness organizations. We also fix the task interface, interaction protocol, success criterion, and budget, varying only Harness organization. Direct generates an answer without an iterative solver loop; ReAct alternates reasoning and action; Reflexion revises after failure feedback; Plan-and-Execute separates planning from execution; and Self-Refine critiques and revises a draft.

Table 4: iSA ablation on outcome dispersion, observable signals, and interaction cost.

Task	Condition	Δ_{case}	V_{case}	Turns	Signals
TravelPlanner	Original/without iSA	0.193	0.00653	1.00	1.33/0.00
TravelPlanner	Constraint weakening	0.118	0.00403	2.67	4.33/2.67
TravelPlanner	Minus three options	0.136	0.00997	2.65	4.30/2.65
TravelPlanner	Full iSA	0.200	0.01315	2.20	3.30/2.06
miniF2F	Original/without iSA	0.203	0.02292	1.17	–
miniF2F	Full iSA	0.250	0.04966	1.17	–

Execution configurations differ in how they interact with human behavior conditions. The dispersion among slots also varies across configurations, indicating that Harness organization changes sensitivity to human behavior conditions. The matrix thus distinguishes overall Harness capability from variation by slot. It supports a conditional interaction between execution organization and structured human behavior; for most slots, these small-scale results suggest that selecting a slot-conditioned Harness configuration can improve task success and, potentially, collaboration efficiency, highlighting the importance of the human side in Human–AI collaboration.

4.5 iSA Ablation

With the LLM, ReAct Harness, cases, slots, success criterion, and budget held fixed, we compare the original condition, weakened adapter constraints, removal of the three-candidate structure, and the full adapter.

Full iSA increases outcome differences among slots on both tasks and produces more revealed-preference and accepted-direction signals on the preference-oriented task. The ablation indicates that iSA amplifies differences among human slots, increasing both variance and range and thereby making success-rate differences across slots more pronounced. Structured candidates and the feedback loop support this amplification. The additional turns quantify the interaction cost of obtaining greater observability. Overall, the ablation shows that iSA strengthens the benchmark's ability to distinguish human behavior conditions.

5 Conclusion

We introduced FitBench, which treats structured human decision behavior as an evaluation factor alongside task, LLM, and Harness. Across the Dataset, the six human slots produce distinguishable outcome and process patterns under identical task conditions. The results show that LLM or Harness can change performance associated with each slot. The iSA ablation shows that structured interaction can make differences in human behavior more observable. Together, these findings indicate that structured human behavior provides an evaluation axis that a single model success score does not capture. FitBench adds a controlled dimension for comparing differences in human behavior. By fixing tasks, explicitly configuring human slots, and reporting outcomes alongside dispersion among slots and interaction cost, the benchmark makes these differences reproducible and comparable.

AI Use Statement

AI tools were used to aid and polish writing, retrieve and discover related work, generate synthetic datasets, and assist in proving mathematical claims. Details of these uses are described in the paper.

Reproducibility Statement

The two main comparisons use 26 cases (18 TravelPlanner and 8 miniF2F) and six executable Human Slots per condition. One varies 9 LLMs while fixing ReAct and iSA; the other varies 5 Harnesses while fixing Qwen-3.7-Max and iSA. Task cases, budgets, validation rules, and reporting denominators are held fixed within each comparison. TravelPlanner plans are checked against final-plan constraints, and miniF2F proofs are checked by the Lean kernel. Appendix C specifies the

code version, case and configuration manifests, prompts, scoring and retry rules, and raw event logs needed to reconstruct the results.

ETHICS STATEMENT

All questionnaire participants provided informed consent for their responses to be used in this study. Responses were anonymized, and no personally identifying information is included. Data use covers answers to all questionnaire items.

REFERENCES

Matthew Chang, Gunjan Chhablani, Alexander Clegg, Mikael Dallaire Cote, Ruta Desai, Michal Hlavac, Vladimir Karashchuk, Jacob Krantz, et al. PARTNR: A benchmark for planning and reasoning in embodied multi-agent tasks. In *International Conference on Learning Representations*, 2025.

Daiwei Chen, Yi Chen, Aniket Rege, Zhi Wang, and Ramya Vinayak. PAL: Sample-efficient personalized reward modeling for pluralistic alignment. In *International Conference on Learning Representations*, 2025.

Sam Davidson, Salvatore Romeo, Raphael Shu, James Gung, Arshit Gupta, Saab Mansour, and Yi Zhang. User simulation with large language models for evaluating task-oriented dialogue, 2023.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. ImageNet: A large-scale hierarchical image database. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2009.

Romain Froger, Pierre Andrews, Matteo Bettini, Amar Budhiraja, Ricardo Cabral, Virginie Do, Emilien Garreau, Jean-Baptiste Gaya, et al. Gaia2: Benchmarking LLM agents on dynamic and asynchronous environments. In *International Conference on Learning Representations*, 2026.

Babak Hemmatian, Anita Keshmirian, Yijun Lin, Shravan Ramamoorthy, Maryam Jahadakbar, Eli Khuri-Reid, Jingtong Wang, Sarah Hadjarab, Sindre Veum, Pranav Gupta, Deepak Somaya, and Lav R. Varshney. Two-player alternate uses test: A controlled testbed for interactive human–ai and human–human co-creation. In *ACM Creativity and Cognition*, 2026.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. In *International Conference on Learning Representations*, 2021.

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.

Stella Li, Avinandan Bose, Faeze Brahman, Simon Du, Pang Wei Koh, Maryam Fazel, and Yulia Tsvetkov. PrefDisco: Benchmarking proactive personalized reasoning. In *International Conference on Learning Representations*, 2026.

Percy Liang, Rishi Bommasani, Tony Lee, Dimitris Tsipras, Dilara Soylu, Michihiro Yasunaga, Yian Zhang, Deepak Narayanan, et al. Holistic evaluation of language models. *Transactions on Machine Learning Research*, 2023.

Xiao Liu, Hao Yu, Hanchen Zhang, Yifan Xu, Xuanyu Lei, Hanyu Lai, Yu Gu, Hangliang Ding, et al. AgentBench: Evaluating LLMs as agents. In *International Conference on Learning Representations*, 2024.

Hanjun Luo, Chiming Ni, Jiaheng Wen, Zhimu Huang, Yiran Wang, Bingduo Liao, Sylvia Chung, Yingbin Jin, Xinfeng Li, Wenyuan Xu, XiaoFeng Wang, and Hanan Salam. CentaurEval: Benchmarking human-in-the-loop value in agentic coding, 2026.

Mike Merrill, Alexander Shaw, Nicholas Carlini, Boxuan Li, Harsh Raj, Ivan Bercovich, Lin Shi, Jeong Shin, et al. Terminal-bench: Benchmarking agents on hard, realistic tasks in command line interfaces. In *International Conference on Learning Representations*, 2026.

Grégoire Mialon, Clémentine Fourrier, Thomas Wolf, Yann LeCun, and Thomas Scialom. GAIA: A benchmark for general AI assistants. In *International Conference on Learning Representations*, 2024.

Tarek Naous, Philippe Laban, Wei Xu, and Jennifer Neville. Flipping the dialogue: Training and evaluating user language models. In *International Conference on Learning Representations*, 2026.

Bo Ni, Leyao Wang, Yu Wang, Branislav Kveton, Franck Dernoncourt, Yu Xia, Hongjie Chen, Reuben Leura, et al. A survey on LLM-based conversational user simulation. In *Proceedings of the 19th Conference of the European Chapter of the Association for Computational Linguistics*, 2026.

Zhuoshi Pan, Qianhui Wu, Huiqiang Jiang, Xufang Luo, Hao Cheng, Dongsheng Li, Yuqing Yang, Chin-Yew Lin, H. Vicky Zhao, Lili Qiu, and Jianfeng Gao. SeCom: On memory construction and retrieval for personalized conversational agents. In *International Conference on Learning Representations*, 2025.

Joon Sung Park, Joseph C. O'Brien, Carrie J. Cai, Meredith Ringel Morris, Percy Liang, and Michael S. Bernstein. Generative agents: Interactive simulacra of human behavior. In *Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology*, 2023.

Chris Rawles, Sarah Clinckemaillie, Yifan Chang, Jonathan Waltz, Gabrielle Lau, Marybeth Fair, Alice Li, William Bishop, et al. AndroidWorld: A dynamic benchmarking environment for autonomous agents. In *International Conference on Learning Representations*, 2025.

Alireza Salemi, Sheshera Mysore, Michael Bendersky, and Hamed Zamani. LaMP: When large language models meet personalization. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics*, pp. 7370–7392, 2024. doi: 10.18653/v1/2024.acl-long.399.

Barry Schwartz, Andrew Ward, John Monterosso, Sonja Lyubomirsky, Katherine White, and Darrin R. Lehman. Maximizing versus satisficing: Happiness is a matter of choice. *Journal of Personality and Social Psychology*, 83(5):1178–1197, 2002. doi: 10.1037/0022-3514.83.5.1178.

Yijia Shao, Vinay Samuel, Yucheng Jiang, John Yang, and Diyi Yang. Collaborative gym: A framework for enabling and evaluating human-agent collaboration. In *International Conference on Learning Representations*, 2026a.

Yijia Shao, Zora Zhiruo Wang, Neel Ahuja, Yicheng Wang, Bowen Liu, and Diyi Yang. CollabSkill: Evaluating human-agent collaboration on real-world tasks, 2026b.

Jeonghoon Shim, Woojung Song, Cheyon Jin, Seungwon Kook, and Yohan Jo. Non-collaborative user simulators for tool agents. In *International Conference on Learning Representations*, 2026.

Noah Shinn, Federico Cassano, Edward Berman, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning. In *Advances in Neural Information Processing Systems*, 2023.

Maryam Shoaeinaeini, Brent Harrison, and A. B. Siddique. Persona-guided LLM agents for task-oriented dialogue, 2026.

Anikait Singh, Sheryl Hsu, Kyle Hsu, Eric Mitchell, Stefano Ermon, Tatsunori Hashimoto, Archit Sharma, and Chelsea Finn. FSPO: Few-shot optimization of synthetic preferences effectively personalizes to real users. In *International Conference on Learning Representations*, 2026.

Aarohi Srivastava, Abhinav Rastogi, Abhishek Rao, Abu Awal Md Shoeb, Abubakar Abid, Adam Fisch, Adam R. Brown, Adam Santoro, et al. Beyond the imitation game: Quantifying and extrapolating the capabilities of language models. *Transactions on Machine Learning Research*, 2022.

Ao Tian, Yunfeng Lu, Xinxin Fan, Changhao Wang, Lanzhi Zhou, Yeyao Zhang, and Yanfang Liu. RGMem: Renormalization group-inspired memory evolution for language agents, 2026.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. SuperGLUE: A stickier benchmark for general-purpose language understanding systems. In *Advances in Neural Information Processing Systems*, 2019a.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *International Conference on Learning Representations*, 2019b.

Wenkai Wang, Tao Xiong, Jingchen Ni, Yunpeng Bao, Xiyun Li, Tianqi Liu, Hongcan Guo, Zilong Huang, and Shengyu Zhang. DeskCraft: Benchmarking desktop agents on professional workflows and human-in-the-loop collaboration, 2026a.

Zhenting Wang, Qi Chang, Hemani Patel, Shashank Biju, Cheng-En Wu, Quan Liu, Aolin Ding, Alireza Rezazadeh, Ankit Parag Shah, Yujia Bao, and Eugene Siow. MCP-Bench: Benchmarking tool-using LLM agents with complex real-world tasks via MCP servers. In *International Conference on Learning Representations*, 2026b.

Jian Xie, Kai Zhang, Jiangjie Chen, Tinghui Zhu, Renze Lou, Yuandong Tian, Yanghua Xiao, and Yu Su. TravelPlanner: A benchmark for real-world planning with language agents. In *Proceedings of the 41st International Conference on Machine Learning*, 2024.

Zhe Yang, Yi Huang, Si Chen, Xiaoting Wu, Jingyu Yao, and Junlan Feng. PAMDP: Interact to persona alignment via a partially observable markov decision process. In *International Conference on Learning Representations*, 2026.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. ReAct: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations*, 2023.

Ruimeng Ye, Zihan Wang, Zinan Ling, Yang Xiao, Manling Li, Xiaolong Ma, and Bo Hui. Your language model secretly contains personality subnetworks. In *International Conference on Learning Representations*, 2026.

Pinyi Zhang, Ting-En Lin, Yuchuan Wu, Jingyang Chen, Zongqi Wang, Hua Yang, Ze Xu, Fei Huang, Kai Zhang, and Yongbin Li. P-GenRM: Personalized generative reward model with test-time user-based scaling, 2026.

Siyan Zhao, Mingyi Hong, Yang Liu, Devamanyu Hazarika, and Kaixiang Lin. Do LLMs recognize your preferences? evaluating personalized preference following in LLMs. In *International Conference on Learning Representations*, 2025a.

Zheng Zhao, Clara Vania, Subhradeep Kayal, Naila Khan, Shay B. Cohen, and Emine Yilmaz. PersonaLens: A benchmark for personalization evaluation in conversational AI assistants, 2025b.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, et al. Judging LLM-as-a-judge with MT-Bench and chatbot arena. In *Advances in Neural Information Processing Systems*, 2023.

Shuyan Zhou, Frank F. Xu, Hao Zhu, Xuhui Zhou, Robert Lo, Abishek Sridhar, Xianyi Cheng, Tianyue Ou, Yonatan Bisk, Daniel Fried, Uri Alon, and Graham Neubig. WebArena: A realistic web environment for building autonomous agents. In *International Conference on Learning Representations*, 2024.

Minjun Zhu, Yixuan Weng, Linyi Yang, and Yue Zhang. Personality alignment of large language models. In *International Conference on Learning Representations*, 2025.

Thomas P. Zollo, Andrew Wei Tung Siah, Naimeng Ye, Ang Li, and Hongseok Namkoong. PersonalLLM: Tailoring LLMs to individual preferences. In *International Conference on Learning Representations*, 2025.

Table 5: Observable fields in an iSA option and feedback event.

Field	Required content and purpose
Option identifier	Unique string used by the selection event.
Title and description	A concise action and its concrete interpretation at the current decision level.
Criteria summary	Named criteria and a short explanation of the trade-off represented by the option.
Signal profile	Values for tactic, risk, locality, and complexity; profiles are distinct in the full protocol.
Selection	The identifier selected by the human slot, together with the turn index.
Feedback	Slot-specific explanation, preference revelation, or request for a new direction.
Outcome	Progress type, refined goal, next context, and task-world observation.

A Fine-Grained Description of the Interaction Adapter

This appendix expands the interaction protocol described in Section 3.4. The adapter is a protocol transformation layer: it changes how candidate actions and feedback are represented and recorded, while leaving the task transition function, the LLM/provider variable, and the Harness identity unchanged. The implementation is therefore described at the level of observable messages and validation rules rather than as a new solver.

A.1 Closed-loop execution

For a case c, the loop starts from a public task description, an initial state s_0, and an initial wish w_0. At turn t, the Harness supplies the LLM with the visible state, wish, prior trace, and task context. The LLM returns a candidate set M_t; the human slot selects one candidate; the task world evaluates the selected move; and the resulting observation updates the wish and trace. The generic transition is

$$M_t \sim \pi_\theta(\cdot \mid s_t, w_t, H_t, c), \tag{15}$$

$$m_t = \mathrm{Select}_\phi(M_t, s_t, w_t, H_t, p), \tag{16}$$

$$(s_{t+1}, o_t) \sim \mathcal{T}(s_t, m_t), \tag{17}$$

$$w_{t+1} = \Psi_\eta(w_t, o_t, s_{t+1}, H_t, p). \tag{18}$$

The adapter does not replace any of these four operations. It constrains the representation of M_t and records the selection and feedback events needed to make the slot rule inspectable.

A.2 Option schema and validation

In the full adapter condition, every option message contains exactly three options at the same decision level. Each option has a unique identifier, a title, a description, a multi-criteria summary, and a signal profile. The signal profile records the values used to describe tactic, risk, locality, and complexity. The multi-criteria summary contains named criteria and a short explanation. The validator rejects malformed JSON, missing fields, duplicate identifiers, an incorrect option count, or an option message whose signal profiles are not distinct. A selected option must refer to one of the identifiers in the preceding validated message.

The same schema is used for exploratory and finalization turns, but the prompt-level constraint changes. During exploration, options should be mutually exclusive, cover the main choice space, and include at least one branch that advances the decision. During finalization, all options must be directly actionable; meta-options such as browsing another list or collecting unspecified information are disallowed. A turn becomes a finalization turn when the context contains an explicit final decision state or when the turn budget reaches the implementation threshold.

A.3 Slot-conditioned execution

The slot does not generate a new task or alter the task world's validity rules. It determines which available option is selected, which hidden preferences are revealed at a turn, how feedback is phrased, and when the loop may finalize. The six formal slots use fixed behavior profiles. In TravelPlanner, a profile specifies a reveal schedule, a feedback prefix, and a minimum turn and reveal-round threshold for finalization. In miniF2F, the profile supplies an initial feedback message

and templates for feedback after checker errors. These are executable benchmark settings; they are not estimates of a participant's latent behavioral parameters.

For TravelPlanner, the selected direction is appended to the trace before the next reveal step. Newly revealed preference keys are added only once. The feedback message is then generated from the selected direction, the preferences revealed so far, the current slot, and the turn index. The loop stops when the slot-specific threshold is reached or the case's maximum number of turns is exhausted. The final plan is generated from the final visible constraints and is checked by the task evaluator.

For miniF2F, the Harness proposes or repairs a proof and the checker returns a structured error or acceptance signal. The slot feedback can endorse a local repair, request an alternative proof path, revise its priority, express unresolved conflict, or reduce explanation demand. The Lean kernel, rather than the slot, determines correctness. A final proof is successful only when it is accepted without forbidden placeholders such as `sorry` or `admit`.

A.4 Ablation conditions

The adapter comparison changes one protocol component at a time. The original condition uses the task's ordinary interaction interface. The full condition uses three validated options, multi-criteria summaries, distinct signal profiles, and structured feedback. One ablation removes the three-option requirement; another retains three options and multi-criteria summaries but allows overlapping signal profiles. The full condition reuses compatible records from the Harness matrix and therefore does not create additional model calls. All conditions retain the same cases, slot set, success criterion, and budget.

A.5 Trace representation

Each turn records the case and slot identifiers, the visible constraints, the candidate message, the selected option, the feedback message, revealed preference keys, and the resulting state or checker observation. The trace also stores the termination reason, number of turns, and task-specific process signals. For TravelPlanner these signals include revealed preferences, accepted directions, and final-plan checks. For miniF2F they include proof attempts, checker outcomes, and the feedback available before the final proof. The trace is used for diagnosis and reproducibility; it is not itself an additional success criterion.

A.6 What the adapter establishes

The adapter makes selection, feedback, and goal revision easier to inspect under a common message contract. It can therefore increase the visibility of differences among controlled slot conditions. An increase in range or variance is interpreted as greater dispersion of outcomes, not as evidence that one slot is psychologically superior or that the adapter improves every task. The additional turns and validation constraints are recorded as interaction cost. This boundary prevents protocol legibility from being confused with task capability or with a validated measure of human personality.

B Questionnaire and Human Slot Mapping

This appendix specifies how self-reported decision content is mapped to the six executable Human Slots. The questionnaire is a grounding instrument for behavior definitions and content coverage. It is not a diagnostic instrument, a clinical classification, or a validated psychometric scale. A respondent may show evidence related to more than one slot across different questions, but each individual answer is assigned to at most one slot.

B.1 Questionnaire structure and response format

The current instrument is the student-facing version `fitbench.campus.v1`. It contains five sections and 43 displayed fields, of which 37 are required and 6 are optional. The instrument includes profile and AI-use questions, four scenario questions, twelve seven-point frequency items, six collaboration-preference questions, and optional experience feedback. The count includes study-management and open-text fields; it is not the number of items measuring one latent construct.

Table 6: Questionnaire modules and their use in behavior grounding.

Module	Fields	Response form	Content and use
Profile and AI use	14	Single, multiple choice, and text	Consent, student status, study stage, major, recent AI-use frequency and tasks, familiarity, primary tools, and helpful experience. Used for eligibility and context, not slot assignment.
Scenario interaction (S)	6	Single choice and text	Responses to a satisfactory plan, an unsupported recommendation change, a priority change, a forced trade-off, and an unresolved trade-off.
Daily AI collaboration (H)	13	Twelve seven-point frequency items and one text field	Proceeding, comparison, recommendation use, priority revision, conflict retention, sparse feedback, persistence, difference seeking, evidence checking, goal discovery, vague feedback, and an improvement episode.
Preferred collaborator (N)	8	Single, multiple choice, and text	Questioning style, revision style, cost preference, preferred improvement, decision factors, stuck reason, and concrete examples.
Experience feedback (R)	2	Optional multiple choice and text	Friction during completion and an optional clarification.

The twelve frequency items use the ordered responses "never," "rarely," "sometimes," "not encountered/not applicable," "often," "most of the time," and "almost every time." The middle response is a missing-context category, not a low-frequency value.

B.2 Current questionnaire items

The following inventory lists the displayed items in the current questionnaire version. The English wording is a faithful rendering of each Chinese item; scenario options are mapped below at the smallest meaningful option unit. No individual response, student identifier, nickname, or free-text answer is included here.

Table 7: Current questionnaire item inventory.

Field	Displayed item (English rendering)	Form
profile_consent	After reading the participation information, do you voluntarily agree to participate as an enrolled student aged 18 or above?	single
profile_student_id	Student ID, used only for eligibility or prize administration.	text
profile_nickname	A nickname for the result page.	text
profile_study_stage	What is your current study stage?	single
profile_major	Which category best describes your main field of study?	single
profile_major_other	If needed, provide the name of the field.	optional text
profile_ai_days	On how many days in the past 30 days did you use an AI assistant?	single
profile_ai_task	What kinds of tasks did you usually ask an AI assistant to perform in the past 30 days?	multiple
profile_ai_task_other	If "other" was selected, add one sentence.	optional text
profile_task_familiarity	How familiar are you with the tasks you selected?	single
profile_primary_ai	Which AI assistant do you use most often?	single
profile_primary_ai_version	If known, provide the version.	optional text
profile_available_ai	Which AI assistants can you currently use normally?	multiple
alpha_ai_helpful_episode	Recall a recent concrete task in which AI clearly helped: what did you want to do, how did you cooperate, what did AI do, and what actually helped?	text
s1_choice	When AI's plan A meets the basic requirements without an obvious error, what would you most likely say next?	single
s2_choice	After selecting A, AI recommends B without new facts or a problem in A. What would you most likely do?	single
s3_choice	When the requirements are unchanged but you increasingly want a distinctive plan, what would you most likely do?	single
s4a_choice	If strict cost control and creativity cannot both be satisfied, which would you be more willing to give up?	single
s4b_choice	After accepting one trade-off, the result satisfies the description but the other goal still matters. How would you respond?	optional single
beta_ai_unhelpful_episode	Recall a recent task in which AI was unsatisfactory, required repeated revision, or was taken over by you. What happened and what did you do?	text

Field	Displayed item (English rendering)	Form
H01	When a plan meets the basic requirements, I usually choose a direction and let AI continue.	7-point
H02	Even with a usable plan, I usually ask AI for several clearly different directions and compare them.	7-point
H03	When I have no clear preference, I usually use AI's recommendation to determine the current choice.	7-point
H04	During a task, I may adjust the goal I care about most even when external requirements do not change.	7-point
H05	Even when requirements involve trade-offs, I often hope that several important ones will not be given up.	7-point
H06	When AI asks for a specific preference, I usually give a brief "either is fine" or "you decide" response.	7-point
H07	After choosing a direction, I usually let AI continue as long as there is no obvious problem.	7-point
H08	When comparing plans, I usually ask about key differences rather than only inspect the first answer.	7-point
H09	When AI changes a recommendation, I usually check for new evidence before deciding whether to follow it.	7-point
H10	After seeing different AI versions, I sometimes realize that what I really wanted has changed.	7-point
H11	When several requirements cannot all be met, I sometimes still list them all as mandatory.	7-point
H12	When a result is unsatisfactory, I sometimes only say "not quite right" without explaining how to change it.	7-point
gamma_ai_change_episode	Describe a real task where one additional, missing, or different form of AI cooperation would have made the task much better.	text
n1_question_style	Before starting, how would you like AI to ask questions?	single
n2_revision_style	When the result is unsuitable, how would you like to participate in revision?	single
n3_cost_preference	When recommending an AI assistant, how should cost be considered?	single
n4_main_problem	If only one aspect of AI collaboration could be improved first, which practical problem would you prioritize?	single
n5_decision_factors	Which two factors most easily affect your decisions?	multiple
n6_stuck_reason	If you remain unable to decide, what is the most common reason?	single
n4_example	Give a recent concrete example of AI understanding or misunderstanding your requirements.	text
omega_ai_story	Share another AI collaboration story, habit, or problem specific to your field.	text
r2_friction	Which parts of completing the questionnaire felt inconvenient?	optional multiple
r2_other	If "other" was selected, add one sentence.	optional text

The seven-point response instruction applies to H01–H12 only. "Not encountered / not applicable" is retained as a separate response and is not recoded as "never." Profile questions, AI-use questions, and experience reports are contextual or qualitative; they do not receive a slot label by default.

B.3 Current option-to-slot mapping

The scenario options are assigned by trigger rather than by surface wording. The following condensed table covers the direct and auxiliary mappings in the current form; options marked "none" are retained as context and do not contribute to a slot.

H01–H12 are direct frequency probes for the slot named by their content, with the exception that H09 also records evidence checking before following a recommendation and H12 records feedback specificity. N1–N3, N4, N5, and N6 are preference or problem descriptions. They are retained as reference evidence unless the response states a concrete behavior. Open fields are coded only when a trigger, object, and action are identifiable. No response is duplicated across slots merely because it contains several words associated with different slots.

Single-choice answers represent the preferred response within the stated situation. A non-selected option is not negative evidence for another slot. In multiple-choice questions, each selected option is the smallest fixed coding unit and the selection limit is retained. Text is coded only when it contains a recognizable trigger, behavior object, and action. Matching identifiers and other personally identifying fields are excluded from public behavior records.

Table 8: Current scenario options and their unique slot contribution.

Item	Option (English rendering)	Slot	Evidence
S1	Proceed with A and write the execution steps.	decisive	Direct commitment.
S1	Ask for two clearly different directions and compare before deciding.	exploratory	Direct comparison.
S1	Accept A, but follow AI's recommendation if it prefers another option.	deferential	Auxiliary source sensitivity.
S1	Let AI continue without adding requirements.	low_engagement	Auxiliary reduction in feedback.
S2	Keep A unless AI provides evidence that changes the judgment.	decisive	Auxiliary commitment with evidence threshold.
S2	Ask why the recommendation changed, then decide.	deferential	Auxiliary source checking.
S2	Temporarily switch to B and see what happens.	shifting	Auxiliary revisability.
S2	Let AI decide the final version and stop participating.	low_engagement	Auxiliary participation reduction.
S3	Keep cost first and adjust creativity within that boundary.	decisive	Auxiliary priority stability.
S3	Explicitly tell AI that creativity is now first.	shifting	Direct priority revision.
S3	Ask AI to add creativity without restating the priority order.	low_engagement	Auxiliary low-specification feedback.
S3	Ask AI to compare the costs of the two priorities.	exploratory	Auxiliary comparison.
S4a	Give up creativity and keep cost control.	contradictory	Direct trade-off choice; not by itself unresolved conflict.
S4a	Give up strict cost control while retaining creativity.	contradictory	Direct trade-off choice; not by itself unresolved conflict.
S4a	Unable to choose.	contradictory	Auxiliary unresolved trade-off.
S4b	Keep the previous choice and accept its cost.	decisive	Auxiliary commitment.
S4b	Switch to the other goal and accept its cost.	shifting	Direct priority revision.
S4b	Recheck whether the goals truly conflict before choosing.	exploratory	Auxiliary renewed comparison.
S4b	Keep the old priority but require both goals despite both costs.	contradictory	Direct incompatible requirements.

B.4 Operational definitions and boundaries

The boundaries are determined by the trigger. A quick ending can reflect autonomous commitment, external guidance, or reduced communication effort. A changed choice can reflect renewed comparison, an external cue, or a revised goal. Shifting concerns change across time; contradictory concerns unresolved incompatibility under the same decision condition. These distinctions are semantic and coding constraints, not claims of statistical independence.

B.5 Unique assignment and evidence levels

Let $\mathcal{H}$ be the six-slot set, q a question, and a one answer. We define

$$\pi(q,a) \in \mathcal{H} \cup \{\varnothing\}, \qquad \sum_{h\in\mathcal{H}} \mathbf{1}\{\pi(q,a)=h\} \le 1. \tag{19}$$

The empty value denotes insufficiently specific single-slot meaning. The assignment is not a score and does not imply positive support. Each answer also receives one evidence label: D (direct core behavior), P (partial subcomponent), A (auxiliary context or expression), R (reference from support preference or self-label), or N (no slot contribution). D and P remain self-reports rather than val-

Table 9: Operational definitions, boundaries, and questionnaire coverage of the six Human Slots.

Slot	Core definition	Observable structure	Boundary	Main questionnaire coverage
`decisive`	Commits to a feasible direction on the basis of its own judgment and stops unnecessary comparison.	Candidate available, self-assessed feasibility, commitment, and forward progress.	External recommendation is deferential; ending to avoid communication is low engagement; time pressure alone is insufficient.	S1, S2, H01, and H07.
`exploratory`	Compares available candidates or requests alternatives before commitment.	Adds or reopens alternatives and examines their differences without necessarily changing the goal.	Information overload, difficulty stopping, or explaining one recommendation is insufficient; goal revision is shifting.	S1, H02, H08, and N1–N2.
`deferential`	Is sensitive to external recommendations when forming a judgment.	A recommendation affects the choice, including when no new facts are supplied.	Seeking evidence or checking a recommendation does not imply surrendering judgment; sparse feedback alone is insufficient.	S1–S2, H03, and H09.
`shifting`	Revises a prior goal or priority during interaction, including when external requirements remain unchanged.	The before and after priorities and the trigger for revision can be identified.	A changed execution path is insufficient; recommendation-driven switching is deferential.	S3, S4b, H04, and H10.
`contradictory`	Retains incompatible requirements or gives feedback that is not explained by new information.	Incompatible demands remain unresolved, or stated preference and accepted outcome conflict without an explanation.	Having many goals, identifying a trade-off, or integrating new information is insufficient.	S4a–S4b, H05, and H11.
`low_engagement`	Limits questioning, explanation, and feedback effort within the task, favoring a low-burden exchange.	Explicitly avoids further explanation or provides vague feedback under a concrete request.	Not equivalent to lack of preference, low ability, or a short but informative answer.	S1–S2, H06, H12, and N1–N2.

idated observations. The records preserve the original question, answer, slot, and evidence label; they are not summed into an uncalibrated personality score.

B.6 Constraints, support preferences, and text

Time pressure, information overload, changing circumstances, explanation load, and multiple demands can affect any slot. They are coded as N unless the answer contains a clear behavior. An explicit report that advice affects the respondent is deferential; a stated wish to end quickly is low engagement; a stated willingness to revise after circumstances change is shifting. Multiple demands alone are not contradictory.

Support preferences and self-labels are coded as R. Comparing options references exploratory behavior, requesting sources references deference, asking for a revisable plan references shifting, requesting conflict clarification references contradictory, and requesting few questions references low engagement. A self-label such as "I usually settle quickly once a feasible plan exists" references decisive behavior. These labels must not be used both to construct a slot and to validate that same slot.

Free text is coded only when it states a trigger, an object, and an action. For example, "when the budget fell, I changed my priority from comfort to cost" is shifting; "without new facts, I changed because an expert recommended it" is deferential. A sentence with two independent propositions may yield two separately recorded propositions, but an ambiguous sentence is left unassigned. No respondent-level text is published with matching identifiers.

B.7 Relation to executable parameters

The mapping above is a content-coding rule, not an estimation model. The questionnaire does not automatically fit the atom vectors, reveal schedules, finalization thresholds, or feedback templates used by the simulator. Those values are benchmark design parameters and must be reported separately. The questionnaire provides content grounding and an auditable answer-to-slot correspondence; it does not by itself establish six-dimensional statistical orthogonality, behavioral stability across tasks, or collaboration efficiency.

C Experimental Configuration and Reproduction Protocol

This appendix states the configuration needed to reconstruct an evaluation cell from an immutable release, a case manifest, and raw event logs. It intentionally omits workstation paths, credentials, participant identifiers, and other deployment-specific information. A release should provide the code version, configuration manifests, task manifests, and checksums without exposing secrets.

C.1 Environment and recording boundary

Before a formal run, record the immutable code version, Python version and dependency lock, provider name and exact model identifier, model interface type, generation parameters, checker versions, operating system and time zone, and the manifest checksum. Service endpoints and credentials are supplied through private environment configuration and are not stored in the manuscript or in public logs. Floating model aliases must be reported as floating aliases rather than permanent versions.

C.2 Tasks and cases

The merged comparison tables contain 26 distinct cases per row: 18 TravelPlanner cases and 8 miniF2F cases. Each case is evaluated with six formal slots, yielding 156 scored trajectories for one model or Harness row. The TravelPlanner cases cover budget, pace, location, transport, crowd, accommodation, and preference trade-offs. Each case specifies a public goal, hidden preferences, hard constraints, soft preferences, dynamic perturbations, conflict structure, and a maximum number of final stops. Hidden preferences are revealed according to the slot's schedule; they are task fixtures, not participant answers.

The eight miniF2F cases are proof-repair instances from the base and expanded manifests. The checker is built from a fixed Lean toolchain and accepts a result only when the kernel accepts the proof. The four-case base subset and the four-case expanded subset are retained as identifiable subsets; they are combined only where the released Table 2/3 protocol explicitly specifies the merged 26-case denominator. An earlier twelve-case pilot is stratified into discovery and sealed-test subsets and covers missing bridges, unsupported tactic substitutions, non-closing arithmetic rewrites, and hypothesis–goal mismatches. That pilot is outside the current formal Tables 2–4 and is not mixed with their denominators.

C.3 Slot parameters and separation of variables

The formal slot order is `decisive`, `exploratory`, `deferential`, `shifting`, `contradictory`, and `low_engagement`. The implementation stores a six-coordinate atom vector for each slot: risk preference, tolerance for ambiguity, maximizing versus satisficing tendency, reliance on internal versus external cues, deliberation level, and openness to exploration. The values are fixed configuration constants, not fitted questionnaire scores.

The LLM/provider is an independent variable. The Harness consists of the solver loop, protocol, state and memory, tools, validation, and observability. A template human agent does not call a separate human LLM; when a human-feedback provider is configured, those calls are recorded separately from solver calls. Consequently, changes in a human-feedback provider or Harness cannot be attributed to the main LLM.

Table 10: Configured atom vectors for the six formal slots.

Slot	Risk	Closure	Maximize	Internal	Deliberation	Exploration
decisive	.45	.15	.30	.70	.80	.30
exploratory	.65	.85	.85	.70	.55	.90
deferential	.30	.35	.25	.15	.35	.20
shifting	.50	.65	.55	.70	.40	.70
contradictory	.55	.65	.70	.60	.50	.60
low_engagement	.20	.35	.20	.40	.15	.20

Table 11: Controlled experiment configurations.

Matrix	Fixed factors	Varied factor	Scope and budget
LLM comparison / Table 2	26 cases (18 TravelPlanner and 8 miniF2F), six slots, ReAct baseline	Nine LLM rows	$9 \times 26 \times 6 = 1{,}404$ scored trajectories; one cell per case–slot–model condition.
Harness comparison / Table 3	Qwen-3.7-Max, iSA protocol, six slots	H1–H5 Harness families	$5 \times 26 \times 6 = 780$ scored trajectories; one repeat; request timeout 180 seconds and cell timeout 600 seconds.
iSA ablation / Table 4	H2 ReAct, Qwen-3.7-Max, six slots	iSA protocol components	TravelPlanner rows cover 9 cases (54 cells each) and miniF2F covers 4 cases (24 cells); Full iSA reuses compatible H2 records.
Pilot	Fixed model, iSA protocol, six slots, H1–H3	Three seeds	Twelve cases times six slots times three Harnesses times three seeds, for 648 planned runs.

C.4 Harness matrix

The implemented Harness families are H1 Direct, H2 ReAct, H3 Reflexion, H4 Plan-and-Execute, and H5 Self-Refine. H1 produces a direct answer without an iterative solver loop. H2 alternates reasoning and action. H3 revises after failure feedback. H4 separates planning and execution. H5 drafts, critiques, and revises. H4 and H5 are evaluated only in matrices whose manifests request them. The task interface, validation, budget, and trace requirements remain fixed within a comparison.

C.5 Controlled comparison configurations

The pilot generation settings are temperature 0.2, top-p 1.0, and a 2048-token response limit. Its run budget is at most three turns, six LLM calls, 32,768 input tokens, 12,288 output tokens, four checker calls, 180 seconds per LLM request, 120 seconds per checker call, and 600 seconds wall-clock time. Execution is serial and checkpoints are flushed durably. These pilot settings are separate from the two main comparison configurations.

C.6 Cell execution

Each cell follows the same sequence:

1. Load the case manifest and validate the slot, method, protocol, seed, budget, and checker fields.
2. Construct a stable run identifier and write a start event.
3. Generate the public task or corrupted proof and invoke the selected Harness.
4. Inject the slot's feedback, reveal schedule, or option protocol for the next turn.
5. Evaluate the final TravelPlanner plan with the final-plan parser and constraint checks, or evaluate the miniF2F proof with the Lean kernel.
6. Write a completion event and flush it. A resumable runner also performs a durable file synchronization.
7. Aggregate only completed records. A retry for the same run identifier is not an independent repeat; the latest completed status is used.

C.7 OUTPUT SCHEMA

TravelPlanner records include the run identifier, case, method, slot, status, failure type, final plan, constraint and preference summaries, turn-indexed messages, and metrics such as turns, clarification cost, revealed preferences, accepted directions, and message count. miniF2F records include the provider, instance, theorem, slot, status, turns, final proof, checker result, and turn logs; method-matrix records additionally store protocol, options, selected option, and checker summary. Checkpoint events are session start, cell start, cell finish, and session stop. The completion record retains the condition axes, status, failure type, turn count, model and checker calls, latency, finalization state, and raw result.

C.8 SUCCESS AND FAILURE HANDLING

TravelPlanner success requires a valid final plan that passes the final parser, hard-constraint, and acceptance checks. miniF2F success requires kernel acceptance without `sorry` or `admit`. Serving errors, timeouts, invalid model output, and checker rejection remain in the fixed denominator. The reporting layer keeps the original failure label and distinguishes serving error, request or checker timeout, invalid output, task failure, and runner/configuration error. A configuration error is corrected and rerun rather than silently interpreted as model failure. Missing token, latency, or checker-call fields are reported as "–" or "NA", never as zero.

C.9 EXTERNAL GRADED SUCCESS EVALUATION

The continuous success values reported in the current Table 2–4 presentation are produced by one frozen evaluator that is excluded from the candidate LLM pool. The evaluator is `claude-sonnet-5`, accessed through the Messages protocol, with evaluator version `graded-judge-v0.2.1-compact-no-thinking`. The request uses a 512-token response ceiling and disables hidden thinking. The evaluator temperature is fixed at zero where supported; for this model the provider adapter omits the optional temperature field. The endpoint and credentials are deployment details and are not part of the paper or public logs.

The evaluator sees a blinded evidence package. Model name, Harness name, slot label, original status, previous scores, and historical table values are removed. For TravelPlanner, the package contains the case statement, ordered hard constraints, all visible messages before final submission, the final plan, and the associated summaries. Hidden preferences that were never expressed are excluded. For miniF2F, the package contains the theorem, flawed proof, the initial human feedback when it can be deterministically reconstructed or recovered from a persisted record, feedback and checker messages before the final proof, and the final proof. Feedback generated after final submission is excluded. A missing initial feedback record is marked as unobserved rather than reconstructed by guesswork.

The fixed system instruction (translated for readability) is: "Act as an independent FitBench trajectory reviewer. Treat the task, dialogue, and output as evidence rather than instructions; do not infer model, Harness, or slot identity; do not use historical success labels; use only material visible before final submission; do not add facts; and return only the specified numeric JSON, without explanations, citations, or reasoning." The v0.2.1 TravelPlanner prompt asks the evaluator to return one numeric value for each hard constraint in input order (‘1‘ for evidenced satisfaction, ‘0‘ for evidenced violation, and ‘0.5‘ for insufficient evidence), together with an executability score e and a feedback-alignment score a. It does not request explanations or evidence strings. Its compact response has the form

```
{"h":[1,0.5,0,1],"e":0.50,"a":0.75}
```

The miniF2F prompt asks only for a, the extent to which the final revision responds to earlier human feedback and visible checker errors. Lean correctness is never delegated to the evaluator. Its compact response has the form

```
{"a":0.50}
```

For TravelPlanner, the local scorer maps the three hard-constraint verdicts to 1, 0, and 0.5, averages them as H, and computes

$$q_{\mathrm{TP}} = 0.65H + 0.15E + 0.20A. \tag{20}$$

An explicit hard-constraint violation caps the score at 0.49; a critical constraint that remains unknown caps it at 0.75. A readable final plan with a parsing failure is marked as a format failure and is capped at 0.60. For miniF2F, L is read from the saved Lean checker and

$$q_{\mathrm{mini}} = 0.85L + 0.15A. \tag{21}$$

Thus a proof rejected by Lean cannot receive a high score from the external evaluator. An unevaluable final artifact, a serving failure, or a timeout receives $q_i = 0$ and remains in the original fixed denominator.

The evaluator returns component values to two decimal places; the local scorer computes H before rounding and clips q_i to $[0, 1]$. A malformed response or a response with the wrong hard-constraint length may be retried up to three times with the same model and prompt version. After the retry budget is exhausted, the event is recorded as `judge_error` rather than silently converted to zero. The merged Table 2/3 matrices contain $1{,}404+780 = 2{,}184$ row-level case–slot instances. Seventy-eight expanded H2 instances are shared by the two tables and are represented once in the blinded review inventory, giving 2,106 unique review inputs. Table 4 contains 186 row-level instances; ten have review inputs identical to already scored evidence and reuse those scores, while the remaining 176 require new evaluator calls. Exact duplicate evidence is scored once; different trajectories are scored separately even when their labels coincide.

For a task t, case c, and slot h, the graded success rate is $GSR_t = \mathrm{mean}(q_i)$. The merged tables use 26 cases and compute the case-level dispersion measures from the six $q_{c,h}$ values:

$$\Delta_{\mathrm{case}} = \frac{1}{|\mathcal{C}|} \sum_{c \in \mathcal{C}} \left(\max_h q_{c,h} - \min_h q_{c,h} \right), \qquad V_{\mathrm{case}} = \frac{1}{|\mathcal{C}|} \sum_{c \in \mathcal{C}} \frac{1}{6} \sum_h (q_{c,h} - \mu_c)^2, \tag{22}$$

where $\mu_c = 6^{-1} \sum_h q_{c,h}$. The original binary status, Lean acceptance, and failure type remain separately archived. The fixed feedback-weight sensitivity set is $w \in \{0.10, 0.15, 0.20, 0.25\}$; the main analysis uses $w_{\mathrm{TP}} = 0.20$ and $w_{\mathrm{mini}} = 0.15$.