GREEDYBM: GREEDY UTILITY DISCOVERY FOR AUTOMATED MODELING OF HUMAN STRATEGIC BEHAVIOR

Anonymous authors
Paper under double-blind review

ABSTRACT

Predicting human strategic behavior requires models that capture systematic departures from individual payoff maximization. Random utility models address this need by separating systematic utility from stochastic influences, but specifying the systematic utility typically demands substantial domain expertise and extensive manual refinement. We introduce GreedyBM, a method that constructs systematic utilities greedily, adding one term at a time, with each term encoding an interpretable behavioral tendency. At each stage, GreedyBM prompts a large language model (LLM) to propose candidate terms based on the current systematic utility and on previously evaluated candidates, and then adds the highest-scoring one. We derive a computable lower bound on the training negative log-likelihood (NLL) reduction achieved in a single stage, together with a multi-stage bound that relates candidate quality to the attainable training NLL. Across three strategic scenarios (the Ultimatum Game, repeated rock–paper–scissors, and continuous double auctions), GreedyBM achieves a lower mean test NLL than both expert-designed models and models discovered by existing LLM-based methods, while consuming substantially fewer tokens during search.

1 INTRODUCTION

In strategic scenarios, players may choose differently in the same situation, and their choices systematically depart from individual payoff maximization. Random utility models (RUMs) capture both while retaining utility maximization as the decision rule (McFadden, 1974; Train, 2009). They decompose total utility into *systematic utility*, which relates the *strategic context* (information observable at the time of choice, such as game history and all players' payoffs) to preferences, and a random shock drawn from a specified distribution to represent unrecorded influences. With this probabilistic decision rule in place, modeling behavior amounts to specifying systematic utility.

Researchers conventionally specify systematic utility through a manual loop: they hypothesize behavioral factors, express them mathematically, and revise the expressions against data. When choices depend on game history and the opponent's actions, at each pass in this loop, experts must determine both which aspects of the interaction matter and how they affect preferences. Large language models (LLMs), which encode world knowledge, can take over this loop, proposing candidate expressions from a description of the scenario and revising them based on evaluation results.

This propose–evaluate–revise loop has discovered model expressions and program components in FunSearch, LLM-SR and BoxLM (Romera-Paredes et al., 2024; Shojaee et al., 2025; Li et al., 2024). AutoBM extends it to strategic behavior, combining LLM-generated behavioral expressions into interpretable models that predict choices better than expert-designed models (Xie et al., 2026). However, each AutoBM candidate is a complete model. In each proposal, the LLM must decide both which expressions enter and how they are combined, and must restate the entire model even when it revises only one part.

We narrow each proposal by separating these two decisions. Following conventional RUM specifications (McFadden, 1974; Train, 2009), we fix how terms combine: systematic utility is a weighted sum of *utility terms*, each a closed-form expression relating an action to its strategic context. The LLM then decides only which term enters next. To reach a good fit with few terms, so that the utility

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.

remains readable, we use a greedy construction that adds at each stage the term that most improves fit (Friedman, 2001; Locatello et al., 2017).

We propose **GreedyBM**, which implements this construction starting from an empty systematic utility. A straightforward way to evaluate a candidate term is to add it to the current systematic utility, refit the model parameters on the training data, and measure the resulting improvement in model fit. Doing so for every candidate would require a separate model refit. GreedyBM instead scores each candidate term against the fixed current model, by measuring how much the current model mispredicts the behavior encoded by that term. Computing the score requires no refitting, and its squared value determines a lower bound on the decrease in training negative log-likelihood (NLL). These scores also guide the LLM's subsequent proposals. After adding the highest-scoring term, GreedyBM refits all weights, so the next stage, scored against the updated model, targets behavior the model still mispredicts.

Our contributions are:

- **Stagewise greedy construction of utility.** We develop an LLM-based method that builds systematic utility one term at a time. Among the evaluated candidates, each stage selects the term whose encoded behavior the current model mispredicts most, measured by the absolute difference between the term's observed and model-predicted averages.
- **A theoretically grounded selection criterion.** We prove that our selection score ranks candidates by guaranteed training NLL reduction. We also establish a multi-stage bound relating attainable training NLL to candidate quality, measured by the selected term's score relative to the best score over all allowed terms.
- **Predictive performance and search cost.** We evaluate GreedyBM on offers in the Ultimatum Game, actions in repeated rock–paper–scissors, and bids in continuous double auctions. GreedyBM achieves lower mean test NLL than expert-designed models and models discovered by AutoBM, BoxLM and FunSearch. Across five runs per scenario, GreedyBM uses 87.2–90.9% fewer search tokens than AutoBM.

2 Related Work

Human choice modeling. Human choice modeling explains observed decisions through hypotheses about preferences and learning (Thaler, 2016). Relative-payoff preferences describe how others' payoffs affect choices (Fehr & Schmidt, 1999; Bolton & Ockenfels, 2000), while learning models describe adaptation to past interactions (Fudenberg & Levine, 1998; Hart & Mas-Colell, 2000). RUMs turn such hypotheses into choice probabilities through systematic utility (McFadden, 1974; Train, 2009). Several expert-designed baselines in Section 4 use predefined expressions for these hypotheses. In contrast, GreedyBM searches for utility expressions directly from data rather than relying on manually designed behavioral specifications.

LLM-based model discovery. LLM-based model discovery iterates between proposing model structures and evaluating the predictions of fitted models. BoxLM, CogFunSearch and GeCCo use these evaluations to guide subsequent proposals (Li et al., 2024; Castro et al., 2025; Rmus et al., 2025). AutoBM applies this approach to strategic behavior, assessing a candidate's expressions and their combination together (Xie et al., 2026). As discussed in Section 1, GreedyBM instead searches over individual utility terms and adds them greedily, making the model search substantially more efficient.

Stagewise modeling and greedy approximation. Stagewise methods build a model by repeatedly adding the component that most improves the current fit. Gradient boosting selects each component from a fixed family of base learners using the negative loss gradient (Mason et al., 1999; Friedman, 2001); RUMBoost applies this idea to utility construction with regression trees (Salvadé & Hillel, 2025). Greedy approximation theory bounds the loss of such procedures even when each selected component is only approximately the best (Locatello et al., 2017; Temlyakov, 2015). Unlike classical stagewise methods with a fixed component family, GreedyBM draws candidate terms from an open-ended space of LLM-proposed expressions. Because the LLM-based search may identify only an approximate best term at each stage, we analyze its performance using greedy approximation theory.

3 Problem Formulation and Method

3.1 Problem Formulation and Fitting Objective

We model the choices of one player role in a strategic scenario, where all players' actions jointly determine outcomes and payoffs. At each decision, the player observes a strategic context x, such as game parameters, past actions and payoffs, or market information, and chooses an action y from a finite set $\mathcal{Y}$; continuous action spaces are discretized onto a grid. Because players may choose differently in the same recorded context, we target the conditional choice probability $q(y \mid x)$.

We express these probabilities through a random utility model (McFadden, 1974; Train, 2009): the player chooses the action that maximizes total utility $U(x, y) = V(x, y) + \varepsilon(x, y)$. The systematic utility V captures preferences shared by players in the same context, while the random shock ε absorbs unrecorded influences. In repeated rock–paper–scissors, for example, a win-stay tendency raises V for repeating a winning action, whereas private beliefs about the opponent's next move enter through ε and lead some players to deviate.

Following standard practice, we assume that, conditional on x, the shocks $\{\varepsilon(x, y)\}_{y\in\mathcal{Y}}$ are independent and follow the standard Gumbel distribution, which yields closed-form choice probabilities (McFadden, 1974). Fixing the scale of ε loses no generality: multiplying all utilities by the same positive constant leaves every choice unchanged, so only the scale of V relative to ε matters (Train, 2009, Ch. 2). The probability that action y maximizes total utility is given by the logit rule:

$$q_V(y \mid x) \triangleq \Pr(U(x, y) \geq U(x, a),\ \forall a \in \mathcal{Y} \mid x) = \frac{\exp V(x, y)}{\sum_{a\in\mathcal{Y}} \exp V(x, a)}. \tag{1}$$

Appendix A.1 extends the method to general shock distributions. Given training data $\mathcal{D}^{\text{train}} = \{(x_i, y_i)\}_{i\in\mathcal{I}^{\text{train}}}$ with $N = |\mathcal{I}^{\text{train}}|$ records, we fit V by minimizing the mean training NLL:

$$\widehat{L}(V) \triangleq -\frac{1}{N} \sum_{i\in\mathcal{I}^{\text{train}}} \log q_V(y_i \mid x_i) = \frac{1}{N} \sum_{i\in\mathcal{I}^{\text{train}}} \left[\log \sum_{y\in\mathcal{Y}} \exp V(x_i, y) - V(x_i, y_i) \right]. \tag{2}$$

To keep V interpretable, we represent it as a weighted sum of *utility terms*. Each term is a closed-form expression $f(x, y; \theta)$ with a behavioral interpretation, possibly combining several behavioral factors and their interactions (Appendix B.1). The term parameters θ govern relationships inside the expression, such as the decay rate of past wins; fixing them yields a function $f \in \mathcal{F}$, where $\mathcal{F}$ is the allowed term space. An external weight λ then scales the term's overall contribution. GreedyBM adds one term per stage and refits all weights at each stage (Section 3.3), so the systematic utility after stage t is

$$V_t(x, y) = \sum_{s=1}^{t} \lambda_s^{(t)} f_s(x, y), \qquad f_s \in \mathcal{F}, \tag{3}$$

where the superscript in $\lambda_s^{(t)}$ indicates that the weight of term s may change as later terms are added. We call V_t, together with the logit rule, the stage-t model. GreedyBM starts from $V_0 = 0$, under which all actions are equally likely, and grows this sum over T stages (Figure 1).

3.2 Selection Score

Each stage produces many candidate terms, so we need a rule for comparing them and deciding which one to add to the current utility V. We define a selection score that determines a lower bound on the training-NLL decrease each candidate can deliver.

Consider adding a term f to V with a scalar weight α. Holding V and the term parameters of f fixed, we differentiate the training NLL with respect to α and evaluate the derivative at $\alpha = 0$:

$$\frac{d}{d\alpha} \widehat{L}(V + \alpha f)\Big|_{\alpha=0} = \frac{1}{N} \sum_{i\in\mathcal{I}^{\text{train}}} \left[\sum_{y\in\mathcal{Y}} q_V(y \mid x_i) f(x_i, y) - f(x_i, y_i) \right] \triangleq \widehat{\Delta}_V(f). \tag{4}$$

Both quantities in brackets are computed from the same term f: the first is its expected value under the predicted action probabilities, and the second is its value at the observed choice y_i. Thus $\widehat{\Delta}_V(f)$

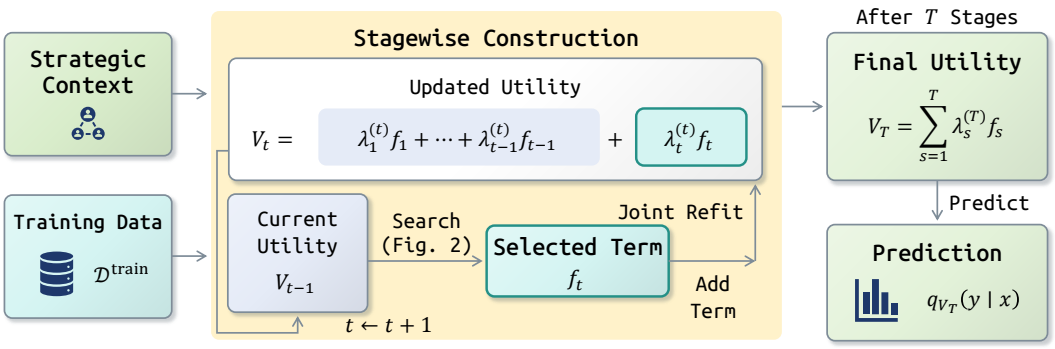


Figure 1: Overview of GreedyBM. In the updated utility, blue marks previously selected terms and green marks the new highest-scoring term. Each stage refits all weights jointly; the final utility produces choice probabilities.

averages predicted minus observed values over the training set, and a nonzero value indicates that the current model mispredicts the behavior f encodes.

As an example, consider the win-stay tendency in Section 3.1. Let $f(x, y)$ equal 1 if the player won the previous round and y repeats the previous action, and 0 otherwise. The observed and predicted averages of f are then the actual and model-expected frequencies of repeating an action after a win. If players repeat more often than the model predicts, then $\widehat{\Delta}_V(f) < 0$; the loss decreases as α increases from 0, so a small positive weight on this term lowers the NLL.

The derivative, however, guarantees a decrease only for infinitesimal steps. To guarantee a decrease at a finite step, we must also bound how quickly the derivative changes, that is, the second derivative. For the logit NLL, the second derivative with respect to α is the average, over records, of the variance of f under the updated action probabilities (Appendix A.6). We therefore require every term in $\mathcal{F}$ to satisfy $|f(x, y)| \le B$ for some $B > 0$, which bounds this variance by B^2 for any V and α. Taylor's theorem then gives the quadratic upper bound

$$\widehat{L}(V + \alpha f) \le \widehat{L}(V) + \alpha\, \widehat{\Delta}_V(f) + \frac{B^2}{2}\alpha^2, \qquad \forall \alpha \in \mathbb{R}. \tag{5}$$

The right-hand side is minimized at $\alpha^\star = -\widehat{\Delta}_V(f)/B^2$. Substituting $\alpha^\star$ shows that the one-step update $V + \alpha^\star f$ lowers the training NLL by at least $[\widehat{\Delta}_V(f)]^2/(2B^2)$. To compare candidate terms using this decrease guarantee, we define the following selection score, whose squared value determines a guaranteed lower bound on the NLL reduction (as shown in Theorem 1):

$$\mathrm{Score}_V(f) \triangleq \frac{|\widehat{\Delta}_V(f)|}{B}. \tag{6}$$

Theorem 1 (Single-stage decrease guarantee)**.** *Under the logit rule with $|f(x, y)| \le B$ for all $f \in \mathcal{F}$, the one-step update for any systematic utility V and term $f \in \mathcal{F}$ satisfies*

$$\widehat{L}(V + \alpha^\star f) \le \widehat{L}(V) - \frac{1}{2}[\mathrm{Score}_V(f)]^2. \tag{7}$$

Ranking candidates by score therefore ranks them by guaranteed decrease.

3.3 Candidate Search and Model Update

To search for a high-scoring term at stage t, we hold V_{t-1} fixed so that all candidates are evaluated against the same model. Each of M search rounds proposes and evaluates a term, adds it to the candidate pool $\mathcal{P}_t$, and feeds the evaluation back to guide the next proposal (Figure 2).

Propose. The LLM receives the task description, available strategic context variables, the current systematic utility, and the representatives produced by the Feedback step below. With probability p_{exp}, we ask it to explore new expression forms; otherwise (exploit), it uses the representatives to

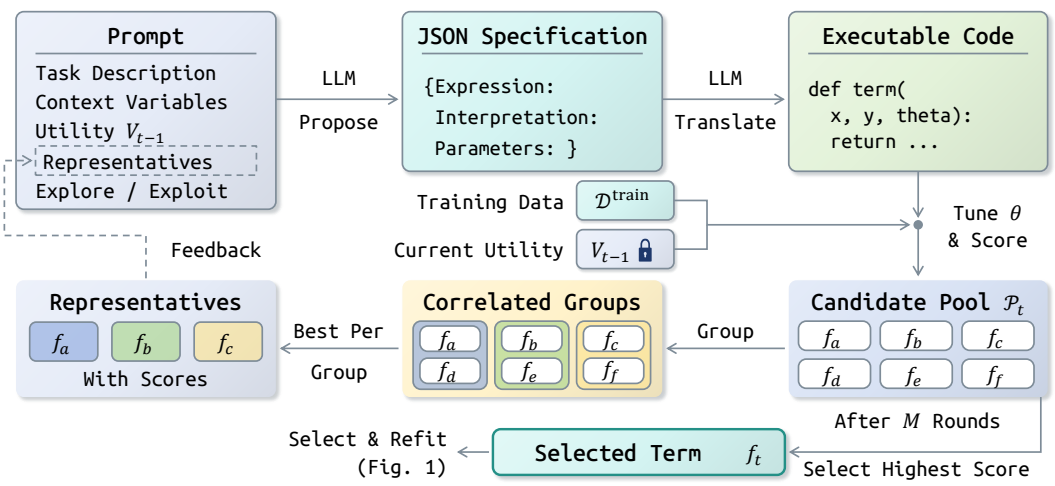


Figure 2: Within-stage search with V_{t-1} fixed. We group candidates by output correlation to obtain prompt examples and select the highest-scoring utility term from the full candidate pool $\mathcal{P}_t$.

improve or combine earlier candidates. In either case, it returns the same JSON specification of the expression, behavioral interpretation, term parameters and allowed ranges, following AutoBM's representation (Xie et al., 2026).

Tune & Score. To evaluate the proposed expression, the LLM translates its JSON specification into PyTorch code. We enforce the output bound of Section 3.2 by transforming the expression g into $f = B\tanh(g/B)$, with $B = 1$ in our experiments; the common bound also prevents a candidate from inflating its score by rescaling its output. We then tune θ within its allowed ranges by maximizing $[\mathrm{Score}_{V_{t-1}}(f(\cdot,\cdot;\theta))]^2$, which has the same maximizers as the score but is differentiable. Finally, we freeze the term parameters and store the term and its score in $\mathcal{P}_t$.

Feedback. Within a stage, the task description and V_{t-1} stay fixed, so later proposals can use earlier search rounds only through feedback. We therefore add evaluated candidates' expressions and scores to subsequent prompts, letting the LLM build on high-scoring expressions and avoid repeated attempts. Different expressions can, however, produce similar outputs, so listing every candidate would concentrate the examples on a few behaviors. Following FunSearch's practice of maintaining distinct search directions (Romera-Paredes et al., 2024), we group candidates whose output vectors over training contexts and actions have Pearson correlation above a threshold, and feed back each group's highest-scoring member as its representative (Appendix B.1; Appendix C.1 evaluates this grouping). Grouping affects only the prompt; stage-end selection uses the full pool $\mathcal{P}_t$.

Select & Refit. After M search rounds, we select the highest-scoring candidate f_t from $\mathcal{P}_t$. The one-step update gives $V_{t-1} + \alpha^\star f_t$, which adjusts only the new weight associated with f_t. Adjusting existing weights as well can lower the NLL further, so we jointly refit all weights:

$$\min_{\boldsymbol{\lambda}\in\mathbb{R}^t} \widehat{L}\left(\sum_{s=1}^{t} \lambda_s f_s\right). \tag{8}$$

This problem is convex in $\boldsymbol{\lambda}$, since each record's NLL is a log-sum-exp of affine functions minus an affine function. We initialize the solver at the one-step update and retain its output only if its NLL does not exceed that of the one-step update, so V_t keeps the decrease in Theorem 1:

$$\widehat{L}(V_t) \le \widehat{L}(V_{t-1}) - \frac{1}{2}[\mathrm{Score}_{V_{t-1}}(f_t)]^2. \tag{9}$$

Appendix B.1 gives the numerical details.

3.4 MULTI-STAGE FITTING GUARANTEE

Equation (9) ties each stage's progress to the selected term's score, but a finite candidate pool may miss the highest-scoring terms in $\mathcal{F}$. We use candidate quality ρ_t to lower-bound the selected term's score relative to the supremum:

$$\mathrm{Score}_{V_{t-1}}(f_t) \geq \rho_t \sup_{f \in \mathcal{F}} \mathrm{Score}_{V_{t-1}}(f), \qquad \rho_t \in (0, 1]. \tag{10}$$

By Theorem 1, the selected term guarantees at least a fraction ρ_t^2 of the best one-step decrease guarantee over the allowed term space.

To assess what these decreases achieve over multiple stages, we compare against reference models that use the same logit rule and finitely many utility terms from $\mathcal{F}$:

$$\bar{V} = \sum_{j=1}^{m} \mu_j h_j, \qquad h_j \in \mathcal{F}, \qquad B \sum_{j=1}^{m} |\mu_j| \leq C, \qquad C > 0. \tag{11}$$

Here, μ_j is the weight of term j, and $B|\mu_j|$ bounds the absolute value of its contribution, so the normalized weight bound C controls the size of the reference model. GreedyBM's weights and number of terms are unrestricted by C.

Since the single-stage guarantee scales with ρ_t^2, we summarize cumulative candidate quality by $A_T \triangleq \sum_{t=1}^{T} \rho_t^2$. Theorem 2 bounds the training NLL after multiple stages.

Theorem 2 (Multi-stage fitting guarantee)**.** *Under the conditions of Theorem 1, let $V_0 = 0$, and suppose that every weight refit exactly attains the minimum in Equation* (8) *at finite weights and that candidate quality satisfies Equation* (10)*. Then, for any reference model satisfying Equation* (11) *and any $T \geq 1$ completed stages,*

$$\widehat{L}(V_T) \leq \widehat{L}(\bar{V}) + \frac{2C^2}{A_T}. \tag{12}$$

Theorem 2 bounds how far GreedyBM's training NLL can remain above the best training NLL of reference models with weight bound C. More stages and better candidates increase A_T, tightening this bound; if $\rho_t \geq \rho > 0$ at every stage, the gap is at most $2C^2/(T\rho^2)$. Appendix A gives the proofs.

4 EXPERIMENTS

Tasks. We call each strategic scenario a task. In the **Ultimatum Game**, a proposer offers a split of a resource to a responder, who can accept or reject; rejection leaves both with nothing. We predict the share offered by the proposer to the responder (y) from the resource (x) (Lin et al., 2020).

In **repeated rock-paper-scissors (RPS)**, two players simultaneously choose rock, paper or scissors over successive rounds. We use both players' previous actions as the strategic context x to predict the focal player's next action y (Komai et al., 2022).

In **continuous double auctions (CDA)**, buyers and sellers post bids and asks for single units over 150 seconds, trading when a bid meets an ask. We predict a buyer's next bid y from the strategic context x: elapsed time, the buyer's previous bids, transaction prices and current bid and ask books. We normalize all prices by the buyer's reservation price (Ikica et al., 2023).

Evaluation. We split each dataset into training, validation and test sets (Appendix B.2). We compare GreedyBM with AutoBM (Xie et al., 2026), BoxLM (Li et al., 2024), FunSearch (Romera-Paredes et al., 2024) and expert-designed models. GreedyBM and the LLM-based baselines use logit choice probabilities. Following AutoBM, each LLM-based baseline has a proposal budget of 200 complete model specifications after initialization; GreedyBM uses $T = 5$ stages with $M = 20$ search rounds per stage. In this performance comparison, $p_{\exp}$ is 0.4, 0.5 and 0.5 for the Ultimatum Game, RPS and CDA, and all LLM calls use DeepSeek-V4 (reasoning disabled). We fit parameters with L-BFGS.

For the Ultimatum Game and CDA, we compute choice probabilities on a 500-point grid over $[0, 1]$, using the nearest grid point to each observed action for NLL. For reference, uniform guessing yields a test NLL of $\ln 500 \approx 6.215$ on this grid and $\ln 3 \approx 1.099$ in RPS. We report mean and standard deviation (Std) over five runs for test NLL, RPS accuracy and CDA bid MSE (Appendix B).

4.1 Behavioral Modeling Performance

To assess prediction on unseen choices, we compare test performance in Table 1. With five utility terms, GreedyBM achieves lower mean test NLL than every baseline in all three tasks, lower CDA bid MSE, and RPS accuracy comparable to AutoBM.

Table 1: Test performance: NLL and CDA MSE (↓), RPS accuracy (↑). LLM-based methods: mean ± Std over five runs. Expert-designed model NLLs are from Xie et al. (2026); names and definitions are in Appendix B.2.

	Ultimatum Game	RPS		CDA	
Method	NLL	NLL	Acc. (%)	NLL	MSE
GreedyBM	**5.121**±.133	**1.070**±.004	43.08±.70	**4.962**±.164	**.0396**±.0055
AutoBM	5.369±.062	1.080±.004	**43.17**±1.33	5.014±.135	.0470±.0013
BoxLM	5.627±.127	1.093±.001	37.58±.87	5.180±.086	.0630±.0080
FunSearch	5.691±.066	1.095±.001	37.02±1.61	5.143±.090	.0581±.0065

Expert-designed models

Ultimatum Game		RPS		CDA	
Model	NLL	Model	NLL	Model	NLL
QRE	6.060	SPNE	1.099	BB	6.037
IA	5.954	RM	1.098	TDB	6.607
ERC	5.977	FP	1.099		
		WSLS	1.095		

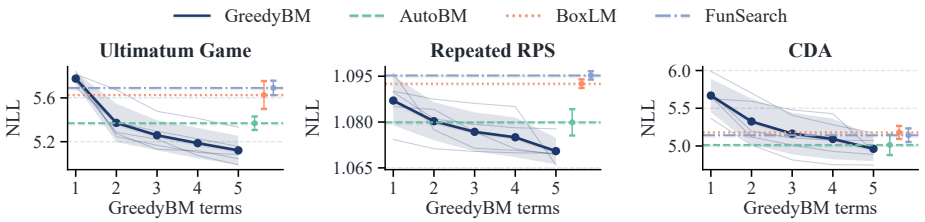


Figure 3: Test NLL across stages. Thin lines show individual GreedyBM runs; thick lines and bands show the mean ±1 Std. Horizontal baseline lines and error bars show five-run means and ±1 Std.

The guarantees in Section 3.4 concern training NLL; Figure 3 tests whether successive additions also improve test performance. Mean test NLL decreases at every stage in all three tasks, alongside improvements in RPS accuracy and CDA bid MSE (Appendix Figure 8). To assess whether this advantage requires more terms, we restrict GreedyBM to the term counts of AutoBM's reported utilities: three in the Ultimatum Game and RPS, five in CDA. GreedyBM retains lower mean test NLL than the AutoBM runs in Table 1 (Appendix Table 3).

4.2 Search Dynamics and Cost

To test whether proposing individual utility terms reduces search cost, we measure the tokens used by the five runs in Table 1. Under the stated proposal budgets, GreedyBM uses 87.2–90.9% fewer tokens than AutoBM (Table 2) while attaining lower test NLL (Table 1).

Proposal counts are not comparable across methods: a GreedyBM proposal is a single term, whereas an AutoBM proposal is a complete model. Tokens provide a common measure of search cost,

Table 2: Total search tokens (thousands): mean ±1 Std over the five runs in Table 1.

Method	Ultimatum Game	RPS	CDA
GreedyBM	**337.9** ± 9.7	**761.2** ± 104.1	**968.7** ± 219.5
AutoBM	3,733.0 ± 87.7	6,959.2 ± 378.0	7,573.7 ± 494.9

so Figure 4 compares the two runs at equal token cost, tracking the best validation NLL in one Ultimatum Game run per method. For each method, we select the run whose validation NLL is closest to its five-run mean.

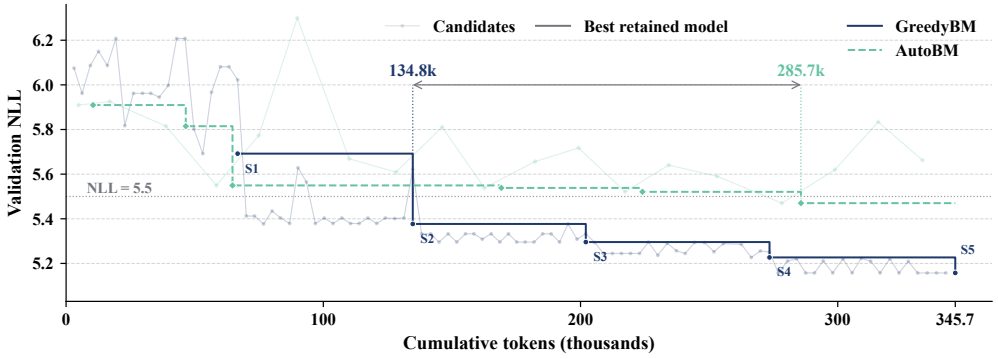


Figure 4: Validation NLL against cumulative search tokens in one Ultimatum Game run per method. Faint curves show evaluated models, including each GreedyBM candidate term with fitted weights. Steps show the best retained model; GreedyBM updates only at stage ends (S1–S5). The horizontal line marks NLL 5.5; the arrow compares the tokens needed to first retain a model at or below this value. Tokens include input and output.

We set the common comparison budget to 345,693 tokens, the cost of completing GreedyBM's five stages. At this budget, GreedyBM reaches validation NLL 5.157 and AutoBM 5.470. GreedyBM first retains a model with validation NLL at or below 5.5 after 134.8k tokens, and AutoBM after 285.7k. AutoBM's best validation NLL plateaus as it evaluates further models, while GreedyBM improves at every completed stage in this run.

4.3 Behavioral Insights from Discovered Models

We trace how successive stages improve predictions of behavior that the preceding model mispredicts, and relate the resulting models to classical theories and AutoBM's findings; Appendix B.3 gives full specifications and the Ultimatum Game analysis.

RPS: from switching to a conditional strategy. Let h_1, h_2 and z_1, z_2 be the focal player's and opponent's latest two actions. The focal player's most frequent action in the last three rounds is m_3, with ties favoring the most recent. Brackets denote indicators, equal to 1 when the enclosed condition holds and 0 otherwise. Here $y \triangleright z$ means y beats z, and W, D, L denote a win, draw and loss in the previous round. The final utility contains five terms, indexed in their order of discovery:

$$V_5(x, y) = \lambda_1^{(5)}[y = m_3] + \lambda_2^{(5)}\big([W/D][y = h_1] + [L][y \triangleright z_1]\big) + \lambda_3^{(5)}[y = \mathrm{S}] + \lambda_4^{(5)}[z_1 = z_2][y \triangleright z_1] + \lambda_5^{(5)}[h_1 \neq h_2][y \notin \{h_1, h_2\}].$$

The terms (1) avoid the recent majority, with $\lambda_1^{(5)} < 0$, (2) repeat after a win or draw and counter the opponent after a loss, (3) favor Scissors, (4) counter a repeated opponent action, and (5) complete the three-action sequence.

The stage-one model penalizes the recent majority action, like AutoBM's frequency penalty (Xie et al., 2026), and treats the other two actions symmetrically. Term 2 then separates them by the

previous outcome. This extends the win-stay example in Section 3 to a form of win-stay–lose-shift (Nowak & Sigmund, 1993). With further terms for Scissors and opponent repetition, the stage-four model predicts a gap between repeat rates after wins and losses of 4.53 percentage points, compared with 4.51 observed.

After two different actions ($h_1 \neq h_2$), the stage-four model assigns probability 36.93% to the unplayed third action, versus 48.45% observed. This *third-action completion* continues the same cyclic direction, a pattern studied by Dyson et al. (2016). The stage-five model captures both completion and outcome-dependent repetition (Figure 5). The aggregate match at 48.45% follows because exact refitting sets $\widehat{\Delta}_{V_5}(f_s) = 0$ for every selected term, equating its predicted and observed averages, including the completion indicator (Proposition 1, Appendix A.4). Predictions within win, draw and loss groups also track observed completion rates (Appendix Table 5).

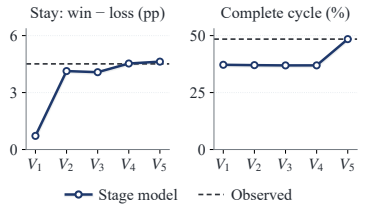


Figure 5: RPS training fit. V_t labels the stage-t model. Left: repeat-rate difference after wins versus losses (percentage points). Right: probability of choosing the unplayed action after two different own actions. Dashed lines show observed values.

When $h_1 \neq h_2$, the recent majority m_3 must be h_1 or h_2, so completion specifies which non-majority action to favor. AutoBM uses decayed and outcome-weighted action counts (Xie et al., 2026); GreedyBM's completion term states explicitly which histories increase the third action's probability.

CDA: bid persistence and upward adjustment. Let y be a normalized bid, h the last own bid, a the lowest ask, b the highest rival bid, and m the market reference price computed from transactions and the order book. Let $r = \max(a, b)$ be the book price and u denote elapsed time.

At stage one, the model favors bids near a target that moves from the last bid h toward the book price r as the deadline approaches:

$$V_1(x, y) = \lambda_1^{(1)} \tanh\left(-\frac{|y - q_1|}{|r - h|}\right),$$
$$q_1 = h + (1 - c)(u/150)^{18.786}(r - h).$$

Here $c = n_\uparrow/(n_\uparrow + n_A + 1)$, where $n_\uparrow$ counts rival bids at least as high as h and n_A counts asks. The target stays near h for most of the trading period because of the large exponent, then moves toward r near the deadline; a larger c reduces this movement.

AutoBM's utility includes a time-dependent tilt (Xie et al., 2026), while V_1 expresses this time dependence through its target price.

GreedyBM then adds a repetition term that concentrates probability near h. The stage-two model approximates the observed repetition rate but still underpredicts upward adjustment (Figure 6).

The remaining terms bring both predicted rates closer to the observed frequencies (Figure 6); Appendix B.3 gives their full expressions and the market-dependent adjustment threshold.

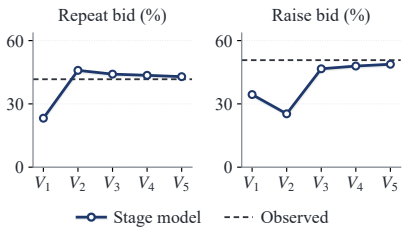


Figure 6: CDA training behavior on 5,125 records. Repeat: $|y - h| < .01$; raise: $y - h > .01$. Dashed lines show observed rates.

5 Conclusion and Limitations

GreedyBM adds closed-form utility terms with fitting guarantees that account for candidate quality. Across three tasks, it achieves lower mean test NLL than expert-designed models and models discovered by AutoBM, BoxLM and FunSearch, using 87.2–90.9% fewer search tokens than AutoBM in these runs. Its terms explain predictions through action-sequence completion and market-dependent bid adjustment. The discovered terms describe regularities of populations observed in controlled studies; whether they persist in field settings, and how they vary across individuals, remain open questions.

AI USE STATEMENT

GreedyBM uses an LLM to propose candidate utility terms and translate them into executable code, as described in Section 3.3. Beyond this use within the method, we used generative AI to check the correctness of theoretical arguments and proofs, draft figures, and draft, translate and revise manuscript text. We did not use generative AI for other research tasks, including formulating theoretical models or mathematical claims, drafting proofs, proposing hypotheses, designing experiments, implementing the method beyond its LLM-generated candidates, generating or processing datasets, or interpreting results. The authors reviewed the AI-assisted material and take responsibility for the final text, mathematical arguments, results and figures.

ETHICS STATEMENT

The evaluation uses previously released behavioral datasets from laboratory or classroom studies.

REPRODUCIBILITY STATEMENT

Appendix A contains the proofs and regularity conditions. Appendix B documents task definitions, fitting procedures and complete utility specifications; Appendix C reports the diversity-log ablation and candidate age replay.

REFERENCES

Mathieu Blondel, André F. T. Martins, and Vlad Niculae. Learning with Fenchel–Young losses. *Journal of Machine Learning Research*, 21(35):1–69, 2020.

Gary E. Bolton and Axel Ockenfels. ERC: A theory of equity, reciprocity, and competition. *American Economic Review*, 90(1):166–193, 2000. doi: 10.1257/aer.90.1.166.

Pablo Samuel Castro, Nenad Tomasev, Ankit Anand, Navodita Sharma, Rishika Mohanta, Aparna Dev, Kuba Perlin, Siddhant Jain, Kyle Levin, Noemi Elteto, Will Dabney, Alexander Novikov, Glenn C Turner, Maria K Eckstein, Nathaniel D. Daw, Kevin J Miller, and Kim Stachenfeld. Discovering symbolic cognitive models from human and animal behavior. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 6849–6890. PMLR, 2025. URL https://proceedings.mlr.press/v267/castro25a.html.

Benjamin James Dyson, Jonathan Michael Paul Wilbiks, Raj Sandhu, Georgios Papanicolaou, and Jaimie Lintag. Negative outcomes evoke cyclic irrational decisions in rock, paper, scissors. *Scientific Reports*, 6:20479, 2016. doi: 10.1038/srep20479.

Ernst Fehr and Klaus M. Schmidt. A theory of fairness, competition, and cooperation. *The Quarterly Journal of Economics*, 114(3):817–868, 1999. doi: 10.1162/003355399556151.

Jerome H. Friedman. Greedy function approximation: A gradient boosting machine. *The Annals of Statistics*, 29(5):1189–1232, 2001. doi: 10.1214/aos/1013203451.

Drew Fudenberg and David K. Levine. *The Theory of Learning in Games*. MIT Press, Cambridge, MA, 1998.

Steven Gjerstad and John Dickhaut. Price formation in double auctions. *Games and Economic Behavior*, 22(1):1–29, 1998. doi: 10.1006/game.1997.0576.

Sergiu Hart and Andreu Mas-Colell. A simple adaptive procedure leading to correlated equilibrium. *Econometrica*, 68(5):1127–1150, 2000. doi: 10.1111/1468-0262.00153.

Barbara Ikica, Simon Jantschgi, Heinrich H. Nax, Diego G. Nunez Duran, and Bary S. R. Pradelski. Competitive market behavior: Convergence and asymmetry in the experimental double auction. *International Economic Review*, 64(3):1087–1126, 2023. doi: 10.1111/iere.12630.

Takahiro Komai, Hiroaki Kurokawa, and Song-Ju Kim. Human randomness in the rock-paper-scissors game. *Applied Sciences*, 12(23):12192, 2022. doi: 10.3390/app122312192.

Michael Y. Li, Emily Fox, and Noah Goodman. Automated statistical model discovery with language models. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 27791–27807. PMLR, 2024.

Po-Hsuan Lin, Alexander L. Brown, Taisuke Imai, Joseph Tao-yi Wang, Stephanie W. Wang, and Colin F. Camerer. Evidence of general economic principles of bargaining and trade from 2,000 classroom experiments. *Nature Human Behaviour*, 4(9):917–927, 2020. doi: 10.1038/s41562-020-0916-8.

Francesco Locatello, Rajiv Khanna, Michael Tschannen, and Martin Jaggi. A unified optimization view on generalized matching pursuit and Frank–Wolfe. In *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, volume 54 of *Proceedings of Machine Learning Research*, pp. 860–868. PMLR, 2017.

Llew Mason, Jonathan Baxter, Peter L. Bartlett, and Marcus R. Frean. Boosting algorithms as gradient descent. In *Advances in Neural Information Processing Systems*, volume 12, pp. 512–518. MIT Press, 1999. URL https://papers.neurips.cc/paper_files/paper/1999/hash/96a93ba89a5b5c6c226e49b88973f46e-Abstract.html.

Daniel McFadden. Conditional logit analysis of qualitative choice behavior. In Paul Zarembka (ed.), *Frontiers in Econometrics*, pp. 105–142. Academic Press, New York, 1974.

Richard D. McKelvey and Thomas R. Palfrey. Quantal response equilibria for normal form games. *Games and Economic Behavior*, 10(1):6–38, 1995. doi: 10.1006/game.1995.1023.

Martin Nowak and Karl Sigmund. A strategy of win-stay, lose-shift that outperforms tit-for-tat in the prisoner's dilemma game. *Nature*, 364(6432):56–58, 1993. doi: 10.1038/364056a0.

Martin J. Osborne. *An Introduction to Game Theory*. Oxford University Press, New York, 2004.

Milena Rmus, Akshay Kumar Jagadish, Marvin Mathony, Tobias Ludwig, and Eric Schulz. Generating computational cognitive models using large language models. In *Advances in Neural Information Processing Systems*, volume 38, Main Conference. Curran Associates, Inc., 2025. doi: 10.52202/085713-2935.

Bernardino Romera-Paredes, Mohammadamin Barekatain, Alexander Novikov, Matej Balog, M. Pawan Kumar, Emilien Dupont, Francisco J. R. Ruiz, Jordan S. Ellenberg, Pengming Wang, Omar Fawzi, Pushmeet Kohli, and Alhussein Fawzi. Mathematical discoveries from program search with large language models. *Nature*, 625(7995):468–475, 2024. doi: 10.1038/s41586-023-06924-6.

Nicolas Salvadé and Tim Hillel. RUMBoost: Gradient boosted random utility models. *Transportation Research Part C: Emerging Technologies*, 170:104897, 2025. doi: 10.1016/j.trc.2024.104897.

Parshin Shojaee, Kazem Meidani, Shashank Gupta, Amir Barati Farimani, and Chandan K. Reddy. LLM-SR: Scientific equation discovery via programming with large language models. In *Proceedings of the International Conference on Learning Representations*, 2025.

V. N. Temlyakov. Greedy approximation in convex optimization. *Constructive Approximation*, 41: 269–296, 2015. doi: 10.1007/s00365-014-9272-0.

Richard H. Thaler. Behavioral economics: Past, present, and future. *American Economic Review*, 106(7):1577–1600, 2016. doi: 10.1257/aer.106.7.1577.

Kenneth E. Train. *Discrete Choice Methods with Simulation*. Cambridge University Press, Cambridge, 2 edition, 2009.

Xiaohan Xie, Haoran Yu, Biying Shou, and Jianwei Huang. Automated human strategic behavior modeling via large language models. *Proceedings of the AAAI Conference on Artificial Intelligence*, 40(21):17877–17885, 2026. doi: 10.1609/aaai.v40i21.38846.

A Proofs and Supporting Results

A.1 Objective Constants, Finiteness, and Shift Invariance

The main text uses independent standard Gumbel shocks. To describe the extension, fix the conditional shock distribution given each strategic context x and write $\boldsymbol{v} = (v_y)_{y\in\mathcal{Y}}$ and $\boldsymbol{V}_x = (V(x,y))_{y\in\mathcal{Y}}$. The expected-maximum surplus is

$$G_x(\boldsymbol{v}) \triangleq \mathbb{E}\left[\max_{y\in\mathcal{Y}}\{v_y + \varepsilon(x,y)\} \mid x\right]. \tag{13}$$

Under the regularity conditions below, choice probabilities satisfy $q_V(y \mid x) = \partial G_x(\boldsymbol{v})/\partial v_y|_{\boldsymbol{v}=\boldsymbol{V}_x}$. The fitting objective is

$$\widehat{L}_G(V) \triangleq \frac{1}{N}\sum_{i=1}^{N}\left[G_{x_i}(\boldsymbol{V}_{x_i}) - V(x_i,y_i)\right]. \tag{14}$$

Adding a constant independent of $\boldsymbol{v}$ to G_x leaves probabilities and loss differences unchanged; for Gumbel shocks we use the shifted surplus defined below, for which $\widehat{L}_G = \widehat{L}$.

Assumption 1 (Surplus regularity)**.** *Let $\mathcal{Y}$ be a finite action set. At every training context, G_x is finite, convex, and differentiable on $\mathbb{R}^{|\mathcal{Y}|}$. It satisfies $G_x(\boldsymbol{v}+c\mathbf{1}) = G_x(\boldsymbol{v}) + c$ for all $\boldsymbol{v}$ and $c\in\mathbb{R}$, where $\mathbf{1}$ is the all-ones vector. Its gradient belongs to the probability simplex, the vectors with nonnegative entries summing to one.*

For the expected-maximum surplus, conditional integrability of the shocks and an absolutely continuous conditional joint distribution ensure a finite surplus and almost-surely unique utility maximizers.

Define the convex conjugate $G_x^*(\boldsymbol{p}) = \sup_{\boldsymbol{v}}\{\langle \boldsymbol{p},\boldsymbol{v}\rangle - G_x(\boldsymbol{v})\}$. For a probability vector $\boldsymbol{p}$, the Fenchel–Young loss $\ell_x(\boldsymbol{v},\boldsymbol{p}) = G_x(\boldsymbol{v}) + G_x^*(\boldsymbol{p}) - \langle \boldsymbol{p},\boldsymbol{v}\rangle$ is nonnegative, with equality precisely when $\boldsymbol{p} = \nabla G_x(\boldsymbol{v})$ (Blondel et al., 2020). Assume $G_{x_i}^*(\boldsymbol{e}_{y_i}) < \infty$ at each observed one-hot action vector $\boldsymbol{e}_{y_i}$. The empirical objective then differs from the mean one-hot loss by a finite constant:

$$\frac{1}{N}\sum_{i=1}^{N}\ell_{x_i}(\boldsymbol{V}_{x_i},\boldsymbol{e}_{y_i}) = \widehat{L}_G(V) + \frac{1}{N}\sum_{i=1}^{N}G_{x_i}^*(\boldsymbol{e}_{y_i}), \qquad \widehat{L}_G(V) \geq -\frac{1}{N}\sum_{i=1}^{N}G_{x_i}^*(\boldsymbol{e}_{y_i}). \tag{15}$$

Thus the empirical loss is finite and bounded below. For the expected-maximum surplus, conditional integrability $\mathbb{E}[|\varepsilon(x,y)| \mid x] < \infty$ for each action is sufficient for finite one-hot conjugates: $G_x(\boldsymbol{v}) \geq v_y + \mathbb{E}[\varepsilon(x,y) \mid x]$ implies $G_x^*(\boldsymbol{e}_y) \leq -\mathbb{E}[\varepsilon(x,y) \mid x]$, while evaluating the conjugate at $\boldsymbol{v} = \mathbf{0}$ gives $G_x^*(\boldsymbol{e}_y) \geq -G_x(\mathbf{0}) > -\infty$.

The constant in Equation (15) leaves directional derivatives and objective differences unchanged. Independent standard Gumbel shocks give expected maximum $\log\sum_y e^{v_y} + \gamma_{\mathrm{E}}$, where γ_{E} is the Euler–Mascheroni constant. We subtract γ_{E} and use the shifted surplus $G_x(\boldsymbol{v}) = \log\sum_y e^{v_y}$. Its conjugate is $G_x^*(\boldsymbol{p}) = \sum_y p_y \log p_y$ on the simplex, with $0\log 0 = 0$. Adding γ_{E} to the surplus subtracts it from the conjugate, leaving the full Fenchel–Young loss unchanged.

Shift invariance follows by differentiating $G_x(\boldsymbol{v}+c\mathbf{1}) = G_x(\boldsymbol{v}) + c$ with respect to $\boldsymbol{v}$: $\nabla G_x(\boldsymbol{v} + c\mathbf{1}) = \nabla G_x(\boldsymbol{v})$. Thus V and its action-centered version $V^\circ(x,y) = V(x,y) - |\mathcal{Y}|^{-1}\sum_a V(x,a)$ induce the same probabilities. Their objective contributions also agree because $\langle \boldsymbol{p},\mathbf{1}\rangle = 1$. This establishes the invariance used in Propositions 1 and 2.

Generalized score and guarantees. Under the regularity and finiteness conditions above, the directional derivative of $\widehat{L}_G$ has the form in Equation (4), using the probabilities induced by the chosen shock distribution. Suppose each term has a second-derivative upper bound $\beta_f > 0$ that holds uniformly over V and α. We replace B^2 by β_f in the one-step formulas, define $\mathrm{Score}_V(f) \triangleq |\widehat{\Delta}_V(f)|/\sqrt{\beta_f}$, and use the reference weight bound $\sum_j |\mu_j|\sqrt{\beta_{h_j}} \leq C$. The following proofs establish both guarantees for $\widehat{L}_G$ with this score and weight bound, under each theorem's remaining conditions. For the logit model, setting $\beta_f = B^2$ recovers the main-text statements.

A.2 Proof of Theorem 1

Fix a finite utility V and a bounded utility term f satisfying the directional smoothness condition in Appendix A.1, and put $\phi(\alpha) = \widehat{L}_G(V + \alpha f)$. Differentiating the finite sample sum gives $\phi'(0) = \widehat{\Delta}_V(f)$. Taylor's theorem and $\phi''(\alpha) \le \beta_f$ give the quadratic upper bound, for every $\alpha \in \mathbb{R}$,

$$\widehat{L}_G(V + \alpha f) \le \widehat{L}_G(V) + \alpha \widehat{\Delta}_V(f) + \frac{\beta_f}{2}\alpha^2.$$

The quadratic attains its minimum at $\alpha_f^\star = -\widehat{\Delta}_V(f)/\beta_f$. Substitution evaluates the decrease explicitly:

$$\begin{aligned} \widehat{L}_G(V + \alpha_f^\star f) &\le \widehat{L}_G(V) - \frac{[\widehat{\Delta}_V(f)]^2}{\beta_f} + \frac{[\widehat{\Delta}_V(f)]^2}{2\beta_f} \\ &= \widehat{L}_G(V) - \frac{1}{2}[\mathrm{Score}_V(f)]^2. \end{aligned} \tag{16}$$

Taking $V = V_{t-1}$ and $f = f_t$, the refitting rule in Section 3.3 compares its output with $V_{t-1} + \alpha_{f_t}^\star f_t$ and therefore preserves the decrease, proving Equation (9). The one-step update belongs to the span of the selected terms, including f_t, so an exact minimum over that span satisfies the same comparison.

For a nonzero scalar c, the rescaled utility term satisfies $\widehat{\Delta}_V(cf) = c\widehat{\Delta}_V(f)$, and $\alpha \mapsto \widehat{L}_G(V + \alpha c f)$ has curvature bounded by $c^2\beta_f$. Taking this corresponding bound preserves the score and the one-step update:

$$\frac{|\widehat{\Delta}_V(cf)|}{\sqrt{c^2\beta_f}} = \frac{|\widehat{\Delta}_V(f)|}{\sqrt{\beta_f}}, \qquad -\frac{\widehat{\Delta}_V(cf)}{c^2\beta_f}(cf) = -\frac{\widehat{\Delta}_V(f)}{\beta_f} f.$$

The guaranteed decrease, half the squared score, is consequently unchanged.

The empirical lower bound in Equation (15) also bounds every candidate score. At V_t, define the supremum candidate score $s_t^\star := \sup_{f \in \mathcal{F}} \mathrm{Score}_{V_t}(f)$. Applying Equation (16) to each $f \in \mathcal{F}$ and taking the supremum gives the following bound, whether or not the supremum is attained:

$$(s_t^\star)^2 \le 2\left(\widehat{L}_G(V_t) + \frac{1}{N}\sum_{i=1}^{N} G_{x_i}^*(\boldsymbol{e}_{y_i})\right) < \infty.$$

If $s_t^\star = 0$, the directional derivative along every term vanishes; linearity gives zero derivative along every finite combination of terms. Convexity then makes V_t a minimum over $\mathrm{span}(\mathcal{F})$. A zero maximum over the proposed candidates establishes only that this candidate pool contains no improving direction; other utility terms in $\mathcal{F}$ may still have positive scores.

A.3 Proof of Theorem 2 and Further Rates

Fix the representation $\bar{V} = \sum_{j=1}^m \mu_j h_j$ in Equation (11). Its normalized weight sum is invariant under $h_j \mapsto c h_j$, $\mu_j \mapsto \mu_j/c$ and the corresponding curvature rescaling, since

$$|\mu_j/c|\sqrt{c^2\beta_{h_j}} = |\mu_j|\sqrt{\beta_{h_j}}, \qquad c \neq 0.$$

Suppose the unconstrained weight fit attains its minimum at a finite weight vector. Proposition 1 then gives $\widehat{\Delta}_{V_{t-1}}(f_s) = 0$ for every $s < t$. By linearity, $\widehat{\Delta}_{V_{t-1}}(V_{t-1}) = 0$; this also holds at $V_0 = 0$. The residual formula extends linearly to finite utility term combinations. Define the fitting gap using $[a]_+ = \max\{a, 0\}$:

$$e_t(\bar{V}) := [\widehat{L}_G(V_t) - \widehat{L}_G(\bar{V})]_+.$$

Convexity bounds the gap by a directional derivative toward the reference model. The zero derivative on the current span eliminates the contribution of the current utility, leaving

$$\begin{aligned} e_{t-1}(\bar{V}) &\le [\widehat{\Delta}_{V_{t-1}}(V_{t-1} - \bar{V})]_+ \\ &\le \sum_{j=1}^m |\mu_j|\,|\widehat{\Delta}_{V_{t-1}}(h_j)| \le C s_{t-1}^\star. \end{aligned} \tag{17}$$

Equations (10) and (9) give a decrease of at least $\rho_t^2(s_{t-1}^\star)^2/2$. Combining this with Equation (17), subtracting $\widehat{L}_G(\bar{V})$, and taking the positive part yields

$$e_t(\bar{V}) \le \left[e_{t-1}(\bar{V}) - \frac{\rho_t^2}{2C^2} e_{t-1}(\bar{V})^2\right]_+. \tag{18}$$

For zero initial error, monotonicity keeps the error zero. Otherwise, at any stage with $e_t(\bar{V}) > 0$, the bracket in Equation (18) is positive. Taking reciprocals and using $(1-u)^{-1} \ge 1+u$ for $0 \le u < 1$ gives $1/e_t(\bar{V}) \ge 1/e_{t-1}(\bar{V}) + \rho_t^2/(2C^2)$. Summing through stage T, or using monotonicity if the error has reached zero, proves the refined initial-value rate:

$$e_T(\bar{V}) \le \begin{cases} 0, & e_0(\bar{V}) = 0, \\ \dfrac{2C^2 e_0(\bar{V})}{2C^2 + e_0(\bar{V}) A_T}, & e_0(\bar{V}) > 0. \end{cases} \tag{19}$$

For $T \ge 1$, this is at most $2C^2/A_T$, proving Theorem 2.

Uniform candidate quality. Under exact refitting, the optimum is no worse than the feasible one-step update, so the refitting rule retains the refitted result. For a fixed reference weight bound C, higher cumulative candidate quality A_T gives a tighter bound on the fitting gap; if $\rho_t \ge \underline{\rho} > 0$ at every stage, then

$$\widehat{L}_G(V_T) - \widehat{L}_G(\bar{V}) \le \frac{2C^2}{T\underline{\rho}^2}.$$

Infimum consequence. Because the comparison holds for every finite representation with the same normalized weight bound $C > 0$, it also gives

$$\widehat{L}_G(V_T) \le \inf_{\substack{m \ge 1,\ h_j \in \mathcal{F},\ \boldsymbol{\mu} \in \mathbb{R}^m \\ \sum_{j=1}^m |\mu_j| \sqrt{\beta_{h_j}} \le C}} \widehat{L}_G\left(\sum_{j=1}^m \mu_j h_j\right) + \frac{2C^2}{A_T}.$$

The reference class contains the zero utility and its loss is bounded below by Equation (15), so this infimum is finite even if no representation attains it.

Assumption 2 (Existence of a finite optimal representation)**.** *The empirical infimum $\widehat{L}_G^\star := \inf_{V \in \mathrm{span}(\mathcal{F})} \widehat{L}_G(V)$ is attained by a finite representation $V^\star = \sum_{j=1}^{m_\star} \mu_j^\star h_j^\star$, with $m_\star \ge 1$ and $h_j^\star \in \mathcal{F}$. Let $C_\star := \sum_{j=1}^{m_\star} |\mu_j^\star| \sqrt{\beta_{h_j^\star}}$.*

Corollary 1 (Objective convergence under finite optimal representation)**.** *Under the conditions of Theorem 2 and Assumption 2, let $e_0^\star = \widehat{L}_G(V_0) - \widehat{L}_G^\star$. If $e_0^\star = 0$, all subsequent stage utilities remain optimal. This includes $C_\star = 0$. If $e_0^\star > 0$, then $C_\star > 0$ and*

$$0 \le \widehat{L}_G(V_T) - \widehat{L}_G^\star \le \frac{2C_\star^2 e_0^\star}{2C_\star^2 + e_0^\star A_T}. \tag{20}$$

If the algorithm can continue through stage $T+1$ under the same conditions, its supremum candidate score satisfies

$$s_T^\star \le \frac{\sqrt{2(\widehat{L}_G(V_T) - \widehat{L}_G^\star)}}{\rho_{T+1}} \le \frac{2C_\star}{\rho_{T+1}} \sqrt{\frac{e_0^\star}{2C_\star^2 + e_0^\star A_T}} \tag{21}$$

when $e_0^\star > 0$; at an optimum $s_T^\star = 0$. Along a continuing run with $\rho_t \ge \underline{\rho} > 0$, the objective gap is $O(T^{-1})$ and $s_T^\star = O(T^{-1/2})$.

Proof. If $C_\star = 0$, positivity of each $\beta_{h_j^\star}$ forces every weight in the chosen optimal representation to be zero, so $V^\star = V_0$. Monotonicity and the definition of the infimum give the initial-optimum branch. For $e_0^\star > 0$, Equation (19) applied to $V^\star$ gives Equation (20). One more stage gives

$$\widehat{L}_G^\star \le \widehat{L}_G(V_{T+1}) \le \widehat{L}_G(V_T) - \frac{1}{2}\rho_{T+1}^2 (s_T^\star)^2.$$

Rearranging and substituting the objective-gap bound proves Equation (21). At a global optimum, the directional derivative along every term vanishes. Finally, $A_T \ge \underline{\rho}^2 T$ gives the stated rates for a continuing run. □

A.4 Moment Matching after Exact Refitting

Proposition 1 (Moment matching after exact refitting)**.** *Under Assumption 1 and empirical one-hot finiteness, let $f_1, \dots, f_t$ be frozen bounded utility terms. Suppose the unconstrained weight fit attains its minimum at a finite vector $\boldsymbol{\lambda}^{(t)} \in \mathbb{R}^t$. Then, for every $s \le t$,*

$$\widehat{\Delta}_{V_t}(f_s) = 0, \qquad \frac{1}{N}\sum_{i=1}^{N}\sum_{y\in\mathcal{Y}} q_{V_t}(y \mid x_i) f_s(x_i, y) = \frac{1}{N}\sum_{i=1}^{N} f_s(x_i, y_i).$$

Proof. Define $J_t(\boldsymbol{\lambda}) = \widehat{L}_G(\sum_{s=1}^{t} \lambda_s f_s)$. Differentiating this finite sum at any finite minimizing weight vector gives

$$0 = \left.\frac{\partial J_t(\boldsymbol{\lambda})}{\partial \lambda_s}\right|_{\boldsymbol{\lambda}=\boldsymbol{\lambda}^{(t)}} = \frac{1}{N}\sum_{i=1}^{N}\left[\sum_{y\in\mathcal{Y}} q_{V_t}(y \mid x_i) f_s(x_i, y) - f_s(x_i, y_i)\right] = \widehat{\Delta}_{V_t}(f_s).$$

The first equality is the unconstrained first-order condition. Rearranging equates each term's predicted and observed averages over the training data. □

A.5 Linear Independence on the Training Context–Action Grid

Proposition 2 (Linear independence of action-centered utility term evaluations)**.** *Assume the empirical setting of Proposition 1. Suppose every unconstrained weight fit preceding a selection attains its minimum at a finite weight vector, and* $\mathrm{Score}_{V_{t-1}}(f_t) > 0$ *with* $\beta_{f_t} > 0$ *at every completed stage $t = 1, \dots, T$. Define the action-centered utility terms by*

$$f_s^{\circ}(x, y) = f_s(x, y) - \frac{1}{|\mathcal{Y}|}\sum_{a\in\mathcal{Y}} f_s(x, a).$$

Their training evaluation vectors $\boldsymbol{f}_s^{\circ} = (f_s^{\circ}(x_i, y))_{i,y}$, for $s = 1, \dots, T$, are linearly independent in $\mathbb{R}^{N|\mathcal{Y}|}$. Equivalently, if $\sum_{s=1}^{T} a_s f_s(x_i, y) = c_i$ for every training record i and every action y, then all $a_s = 0$.

Proof. For any context-only evaluation c_i, the empirical residual pairing is zero because $\sum_y q_V(y \mid x_i) = 1$. Suppose a relation $\sum_s a_s f_s(x_i, y) = c_i$ holds and some coefficient is nonzero. Let r be the largest index with $a_r \neq 0$. Pairing the relation with the empirical residual at V_{r-1} gives

$$a_r \widehat{\Delta}_{V_{r-1}}(f_r) = -\sum_{s<r} a_s \widehat{\Delta}_{V_{r-1}}(f_s) = 0.$$

Proposition 1 gives the last equality; for $r = 1$ the sum is empty. This contradicts the positive score of f_r. Finally, a linear relation among centered evaluation vectors is exactly a relation among the original evaluations with a context-only right-hand side, which proves the equivalence. □

A.6 Log-Partition Derivatives and Directional Smoothness

For the shifted surplus $G_x(\boldsymbol{v}) = \log \sum_{y\in\mathcal{Y}} \exp(v_y)$, direct differentiation gives

$$\left.\frac{\partial G_x(\boldsymbol{v})}{\partial v_y}\right|_{\boldsymbol{v}=\boldsymbol{V}_x} = q_V(y \mid x), \qquad \left.\frac{\partial^2 G_x(\boldsymbol{v})}{\partial v_y \partial v_{y'}}\right|_{\boldsymbol{v}=\boldsymbol{V}_x} = q_V(y \mid x)\mathbb{I}\{y = y'\} - q_V(y \mid x) q_V(y' \mid x).$$

The Hessian is $\mathrm{diag}(\boldsymbol{q}_V(\cdot \mid x)) - \boldsymbol{q}_V(\cdot \mid x)\boldsymbol{q}_V(\cdot \mid x)^{\top}$, the positive semidefinite covariance matrix of a one-hot choice. Along a bounded utility term direction, its quadratic form is a variance:

$$\frac{d^2}{d\alpha^2}\widehat{L}_G(V + \alpha f) = \frac{1}{N}\sum_{i=1}^{N} \mathrm{Var}_{y\sim\boldsymbol{q}_{V+\alpha f}(\cdot\mid x_i)}[f(x_i, y)].$$

Popoviciu's inequality bounds this variance by one quarter of the squared range. If $|f(x,y)| \le B_f$ for a finite $B_f > 0$, then for every V and α,

$$\frac{d^2}{d\alpha^2}\widehat{L}_G(V+\alpha f) \le \frac{1}{4N}\sum_{i=1}^{N}\left[\max_{y\in\mathcal{Y}} f(x_i,y) - \min_{y\in\mathcal{Y}} f(x_i,y)\right]^2 \le B_f^2.$$

Thus $\beta_f = B_f^2$ is a valid global directional smoothness constant, and a common output bound gives $\beta_f = B^2$.

B Implementation and Experimental Details

B.1 Utility Term Search and Numerical Fitting

Additive representation. In the weighted sum $V = \sum_s \lambda_s f_s$, each term can combine several behavioral factors, including their interactions. A complete closed-form utility has a one-term representation whenever a rescaled copy belongs to $\mathcal{F}$; the allowed expressions and output bounds determine this inclusion. A new term with zero weight leaves the current model unchanged. Exact refitting over the expanded set of weights can therefore only maintain or improve training fit; the refitting rule preserves the one-step guarantee with a numerical solver.

We run $T = 5$ stages with $M = 20$ search rounds per stage, each completed after a candidate has been successfully translated, fitted and evaluated. Failed code translations do not count toward M. We return the current utility if no candidate has a positive score.

In each prompt, we express the current utility through its selected expressions and symbolic weights.

A parameterized candidate. Figure 7 illustrates how we turn an LLM proposal into a fitted utility term. In this Ultimatum Game example, the proposed expression $g(y) = -|y - r|^\alpha$ has a fixed reference share $r = 0.5$ and a parameter α controlling the penalty for deviations from an equal split. After translating the expression into PyTorch, we fit α from its initial value 1 to 0.2427 and evaluate $f(y) = \tanh(g(y))$. The next prompt includes this candidate and its score as an evaluated example.

Our grouped feedback follows FunSearch's use of distinct search groups (Romera-Paredes et al., 2024). To group terms by their behavior, we construct a signature for each term from its fitted, bounded outputs on training contexts and the action set $\mathcal{Y}$. For the Ultimatum Game, we flatten these outputs into one vector. For sequential tasks, we partition contexts by history length into buckets 0, 1, 2–3, 4–7, and so on; in CDA, we use own-bid history length and merge lengths of 16 or more. When there are more than 2,000 contexts, we sample proportionally within these buckets and keep the sample fixed throughout the stage. We compute Pearson correlation between two signatures within each bucket and average the correlations across buckets, retaining their signs.

We omit buckets with fewer than five scalar entries; numerically equal vectors have similarity one, and unequal vectors with a zero-variance member have similarity zero. If no bucket remains, we assign similarity one.

We process candidates in evaluation order and compare each signature only with the current group representatives. If the highest correlation exceeds the threshold, we add the candidate to that group; otherwise, we create a new group. We retain the highest-scoring candidate in each group as its representative. The correlation thresholds are 0.85 for the Ultimatum Game, 0.70 for RPS and 0.65 for CDA.

Numerical fitting. We fit parameters with L-BFGS, falling back to Adam on failure. Each translated expression g is transformed into $f = B\tanh(g/B)$ with $B = 1$; only candidates with finite outputs and scores enter the pool. For the stage-t weight refit, let $\alpha_t^\star = -\widehat{\Delta}_{V_{t-1}}(f_t)/B^2$. We initialize the existing weights at their previous values and the new weight at $\alpha_t^\star$, giving the one-step update. A numerical solver may not attain a minimum at finite weights. We accept its output only when all weights are finite and the NLL is no larger than at initialization; otherwise, we retain the one-step update. This rule preserves Theorem 1; Theorem 2 assumes that each unconstrained weight fit attains its minimum at a finite weight vector.

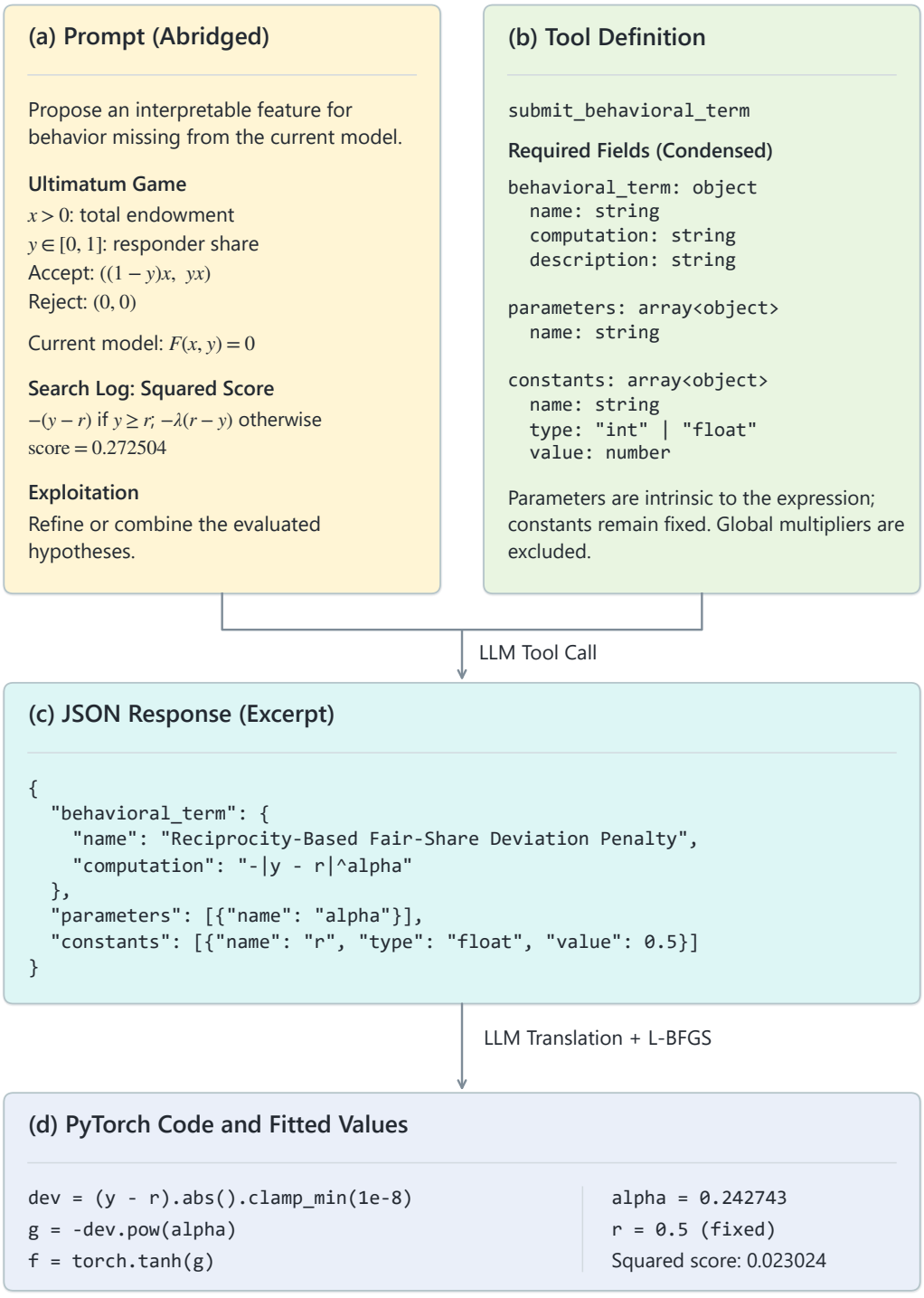


Figure 7: An Ultimatum Game candidate from prompt to fitted utility term. The prompt, tool definition and PyTorch code are condensed, and the JSON response omits its description field. The prompt retains its original terminology and notation, which are independent of the main text. Scores are reported as Score^2.

B.2 Tasks, Baselines and Reported Statistics

Data. The Ultimatum Game dataset has 38,287 records with a 70/15/15 training, validation and test split (Lin et al., 2020; Xie et al., 2026). RPS contains 500 episodes of 50 rounds (Komai et al., 2022). CDA contains 17,237 buyer bids, normalized by the buyer's reservation price (Ikica et al., 2023).

Action grids. For the Ultimatum Game and CDA, we use the 500-point action grid $\mathcal{Y} = \{j/499 : j = 0, \ldots, 499\}$. To compute NLL, we map each observed action to the nearest grid point: $\widetilde{y}_i = j_i/499$, where $j_i = \arg\min_{j\in\{0,\ldots,499\}} |y_i - j/499|$, and evaluate $-\log q_V(\widetilde{y}_i \mid x_i)$. We resolve equal computed distances by choosing the smaller index. CDA context prices retain their original normalized values; we compute bid MSE against the original observed bids.

Data splits and repetitions. For the sequential tasks, we assign complete RPS episodes and CDA market trajectories to the training, validation and test sets, constructing each history within its trajectory. Across the five runs, we keep the data splits and random-seed settings fixed; variation comes from stochastic LLM generation.

Expert-designed models. We give the expert-designed specifications reported in the supplement of Xie et al. (2026). For a utility-based specification V_θ, the logit choice rule is

$$q_\theta(y \mid x) = \frac{\exp V_\theta(x, y)}{\sum_{a\in\mathcal{Y}} \exp V_\theta(x, a)}.$$

Below, $[v]_+ = \max(v, 0)$, and parameter symbols are local to each model.

Ultimatum Game

Let x be the resource and $y \in [0, 1]$ the responder share. If the offer is accepted, the proposer and responder receive $x(1 - y)$ and xy, respectively.

Quantal response equilibrium (QRE). QRE assigns stochastic choices according to expected payoff (McKelvey & Palfrey, 1995). With $P_{\mathrm{acc},\theta}(y \mid x)$ denoting the responder's acceptance probability, the reported specification is

$$P_{\mathrm{acc},\theta}(y \mid x) \propto \theta_2 xy, \qquad V_\theta(x, y) = \theta_1 x(1 - y)P_{\mathrm{acc},\theta}(y \mid x).$$

Here θ_1 controls the proposer's sensitivity to expected payoff, and θ_2 enters the responder's acceptance response.

Inequity aversion (IA). IA assigns separate penalties to disadvantageous and advantageous inequality (Fehr & Schmidt, 1999):

$$V_\theta(x, y) = \theta_1 x(1 - y) - \theta_2[x(2y - 1)]_+ - \theta_3[x(1 - 2y)]_+.$$

The coefficients $\theta_2, \theta_3 \geq 0$ penalize receiving less and more than the responder, respectively.

Equity, reciprocity and competition (ERC). The ERC specification combines own payoff with a quadratic penalty for deviations from equal division (Bolton & Ockenfels, 2000; Xie et al., 2026):

$$V_\theta(x, y) = \theta_1 x(1 - y) - \theta_2(y - 1/2)^2.$$

The two coefficients weight self-interest and fairness.

Repeated Rock-Paper-Scissors

Let k index rounds, x_k contain both players' actions before round k, and $r(y, z)$ denote the stage-game payoff from action y against the opponent's action z.

Subgame perfect Nash equilibrium (SPNE). The equilibrium baseline assigns equal probability to the three actions (Osborne, 2004):

$$q(y \mid x_k) = \tfrac{1}{3}, \qquad y \in \{\mathrm{Rock}, \mathrm{Paper}, \mathrm{Scissors}\}.$$

Regret matching (RM). Let $R_k(y)$ denote cumulative payoff regret for action y based on the preceding rounds. RM normalizes its positive part (Hart & Mas-Colell, 2000):

$$q(y \mid x_k) = \begin{cases} [R_k(y)]_+ / \sum_{a \in \mathcal{Y}} [R_k(a)]_+, & \sum_{a \in \mathcal{Y}} [R_k(a)]_+ > 0, \\ 1/3, & \text{otherwise.} \end{cases}$$

Fictitious play (FP). FP estimates the opponent's action probabilities from historical frequencies (Fudenberg & Levine, 1998). If $N_{k-1}(z)$ counts action z before round k, then for $k > 1$,

$$b_k(z) = \frac{N_{k-1}(z)}{k-1}, \qquad V_\theta(x_k, y) = \theta \sum_{z \in \mathcal{Y}} b_k(z) r(y, z).$$

The reported baseline uses the logit rule above, with θ controlling sensitivity to expected payoff (Xie et al., 2026).

Win-stay, lose-shift (WSLS). Let h be the previous own action and o the previous outcome. The reported WSLS specification uses fitted stay and shift probabilities $p_{\text{stay}}, p_{\text{shift}} \in [0, 1]$ (Nowak & Sigmund, 1993; Xie et al., 2026):

$$q(y \mid x_k) = \begin{cases} p_{\text{stay}}, & o = \text{win},\ y = h, \\ (1 - p_{\text{stay}})/2, & o = \text{win},\ y \neq h, \\ 1 - p_{\text{shift}}, & o \neq \text{win},\ y = h, \\ p_{\text{shift}}/2, & o \neq \text{win},\ y \neq h. \end{cases}$$

Non-wins include draws; switching probability is split equally between the two alternative actions.

Continuous Double Auctions

Let y be a bid normalized by the buyer's reservation price and u elapsed time. The normalized profit from an accepted bid is $1 - y$.

Belief-based bidding (BB). BB estimates acceptance from transaction and order-book records (Gjerstad & Dickhaut, 1998). Let $\mathcal{T}_x$ contain transaction prices, $\mathcal{A}_x$ ask prices, and $\mathcal{B}_x$ bid prices. Define the counts

$$C_<(x, y) = \sum_{p \in \mathcal{T}_x} \mathbb{I}\{p < y\} + \sum_{a \in \mathcal{A}_x} \mathbb{I}\{a < y\},$$
$$C_>(x, y) = \sum_{b \in \mathcal{B}_x} \mathbb{I}\{b > y,\ b \notin \mathcal{T}_x\}.$$

The reported acceptance belief and expected-profit utility are

$$P_{\text{acc}}(y \mid x) \propto \frac{C_<(x, y)}{C_<(x, y) + C_>(x, y)}, \qquad V_\theta(x, y) = \theta(1 - y) P_{\text{acc}}(y \mid x),$$

with the logit rule converting expected profit into choice probabilities.

Time-dependent bidding (TDB). TDB specifies a point bid as a function of elapsed time (Xie et al., 2026):

$$\widehat{y}(u) = \sigma\left(\theta_1 + \theta_2 e^{\theta_3 u}\right), \qquad \sigma(v) = \frac{1}{1 + e^{-v}}.$$

The parameters determine the level and time dependence of the bid trajectory.

B.3 Stagewise Utilities and Behavioral Predictions

We analyze one run per task from one through five selected utility terms (Section 4.3), using 26,800 Ultimatum Game offers, 17,500 RPS records and 12,065 CDA bids from the training data. All displayed weights are fitted on training data. We weight aggregate Ultimatum Game predictions by the observed resource distribution; 23,001 of the 26,800 offers have resource 100.

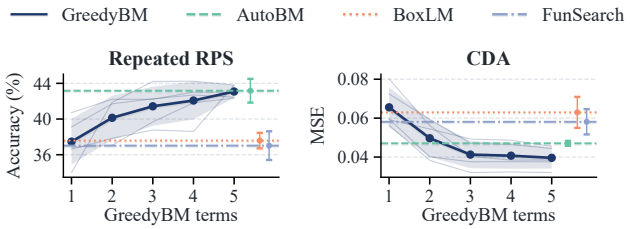


Figure 8: RPS accuracy and CDA bid MSE across stages on the test set. Thin lines show individual GreedyBM runs; thick lines and bands show the mean ± 1 Std. Horizontal baseline lines and error bars show five-run means and ± 1 Std.

Table 3: Test NLL at GreedyBM stages matching the term counts of the AutoBM utilities reported by Xie et al. (2026). Values are five-run means.

Task	Stage / reported terms	GreedyBM	AutoBM
Ultimatum Game	3	5.259	5.369
RPS	3	1.077	1.080
CDA	5	4.962	5.014

RPS specification. Let y be a candidate action, h_1, h_2 the focal player's two most recent actions, z_1, z_2 the corresponding opponent actions, and $o_1 \in \{-1, 0, 1\}$ the latest outcome. With Rock, Scissors and Paper encoded as 0, 1 and 2, $y \triangleright z$ means $y = (z + 2) \bmod 3$. Here $B = 1$; absorbing each indicator's $\tanh(1)$ factor into its weight gives

$$V_5 = -0.189275 I_m + 0.124459 I_o + 0.096053 I_s + 0.076770 I_r + 0.577284 I_c.$$

Indicator	Behavioral condition
I_m	y is the most frequent own action in the latest three available rounds; ties favor the most recent occurrence.
I_o	After a win or draw, $y = h_1$; after a loss, $y \triangleright z_1$. In the first round, $y = \text{Rock}$.
I_s	With at least one previous round, $y = \text{Scissors}$.
I_r	The opponent's last two actions agree, $z_1 = z_2$, and $y \triangleright z_1$.
I_c	The last two own actions differ, $h_1 \neq h_2$, and y is their remaining third action.

Indicators requiring unavailable history are zero, except for the first-round branch of I_o.

Table 4: RPS stage weights in the indicator basis and training NLL.

Stage	I_m	I_o	I_s	I_r	I_c	NLL
1	$-.372615$	—	—	—	—	1.084891
2	$-.410144$	.111526	—	—	—	1.083690
3	$-.410550$	.112668	.054808	—	—	1.083364
4	$-.402631$	.090950	.055721	.080935	—	1.083093
5	$-.189275$	.124459	.096053	.076770	.577284	1.060019

For Figure 5, we evaluate the win–loss difference in next-action repetition on histories with a previous outcome, and third-action completion on the 12,144 records with $h_1 \neq h_2$. Predictions average the complete choice distribution over the same recorded contexts as the observed rates. The predicted win–loss repeat gaps at stages one through five are 0.72, 4.13, 4.07, 4.53 and 4.63 percentage points, compared with 4.51 observed. Table 5 reports completion rates by previous outcome.

CDA specification. Let $D_k(v) = v$ for $v \geq 0$, kv otherwise, and let u denote elapsed time. Normalize prices by the buyer's reservation price. Let h be the last own bid (zero if absent), a

Table 5: RPS third-action completion after two different own actions, by previous outcome. Predictions use the stage-five model on training contexts.

Previous outcome	Predicted (%)	Observed (%)
Win	47.2	47.1
Draw	47.9	46.9
Loss	50.2	51.2

the lowest ask (one if absent), b the highest rival bid (zero if absent), and $r = \max(a, b)$ the book price. Let $n_\uparrow$ count rival bids at least as high as h, n_B rival bids, n_A asks, and n_Q all active bids including own bids. The market reference price m averages the mean transaction price and the book midpoint when both are available, uses the available one otherwise, and falls back to h when neither is available. The midpoint uses the highest active bid and lowest ask.

With unit output bounds, the selected utility terms are $f_j = \tanh(g_j)$. The final utility, with rounded parameters, is

$$\begin{aligned} V_5(x, y) = {} & 2.80303 \tanh(g_1) + 1.81563 \tanh(g_2) - 0.59934 \tanh(g_3) \\ & - 0.85422 \tanh(g_4) - 0.27296 \tanh(g_5). \end{aligned} \tag{22}$$

The first three raw expressions are

$$\begin{aligned} c &= n_\uparrow/(n_\uparrow + n_A + 1), \qquad q_1 = h + (1 - c)(u/150)^{18.78595}(r - h), \\ g_1 &= \; -|y - q_1|/|r - h|, \\ q_2 &= a + [n_\uparrow/(n_B + 1) - .91258]_+(a - h), \\ g_2 &= 8.95123\mathbb{I}\{|y - h| < .01\} - |y - q_2| - 32.65054[y - r]_+^2, \\ \zeta &= m + 94.41370(h - m), \qquad g_3 = -\big[|y - \zeta| + D_{98.93419}(y - m)\big]. \end{aligned}$$

The g_1 expression applies when $|r - h| \geq 10^{-8}$; otherwise it is action-constant. The displayed g_2 applies before the deadline ($u < 150$).

For the fourth term, let d be the change from first to last transaction divided by the number of intervening transactions, or zero with fewer than two, and let E be a clipped exponential. With rival bids, $\ell_4 = \min(a, b - 5.39325)$; otherwise $\ell_4 = a$. The reposting indicator requires own history, $h < a$, and $|y - h| < 10^{-6}$. Define

$$\begin{aligned} w &= E\big(-(150 - u)_+/(10 \times 90.01618)\big), \quad \widetilde{d} = \mathrm{clip}\big(d/\max(a, 10^{-6}), -1, 1\big), \\ s &= \max(m, h) + w\widetilde{d}(a - h), \quad p = (1 - n_B/(n_B + n_A + 1))(1 - w), \\ q_4 &= wh + (1 - w)[p\ell_4 + (1 - p)s], \quad A_Q = n_Q/(n_Q + n_A + 1), \\ g_4 &= \; -\big[|y - q_4| + D_{15.14796}(y - m) + w[y - h]_+ + (1 - w)[b - y]_+ \\ &\qquad + w[d]_+ + \tfrac{1}{2}(1 - w)[s - y]_+ - 92.72112[y - a]_+^2 \\ &\qquad + 103.51104(1 - A_Q)I_{\mathrm{repost}}\big]. \end{aligned}$$

For the fifth term, $\ell_5 = \min(a, b + 1.08517)$ with rival bids and $\ell_5 = a$ otherwise. The quantity ξ averages two indicators: negative transaction-price drift and the last transaction exceeding the transaction mean. The stale-quote indicator requires own history, $h < a$, and $|y - h| < 10^{-8}$. Define

$$\begin{aligned} w_5 &= E\big(-(150 - u)_+/(10 \times 10^{-8})\big), \\ r_5 &= w_5\{1 - \mathrm{clip}[(a - h)/(|h| + 10^{-8}), 0, 1]\}\xi, \\ q_5 &= w_5[h - r_5\{h - \tfrac{1}{2}(b + \tfrac{1}{2}a)\}]_+ + (1 - w_5)\ell_5, \\ g_5 &= \; -\big[D_{80.81835}(y - q_5) + 1.21501[y - a]_+^2 + 101.72487(1 - r_5)I_{\mathrm{stale}}\big]. \end{aligned}$$

Table 6: CDA stage weights and training NLL.

Stage	λ_1	λ_2	λ_3	λ_4	λ_5	NLL
1	4.218007	—	—	—	—	5.425285
2	1.842777	1.891320	—	—	—	5.063798
3	1.716188	2.293089	−.903918	—	—	4.872782
4	2.730467	1.822955	−.608816	−.932686	—	4.815108
5	2.803030	1.815632	−.599340	−.854222	−.272956	4.812317

For Figure 6, we use the 5,125 records from 13 markets with own-bid and ask history, $0 < h < a < 1$, and $u < 150$. Repeat means $|y - h| < .01$, and raise means $y - h > .01$. We evaluate every stage on these same records by integrating its 500-point choice distribution.

On 91.6% of all training records, $h < m$ and $\zeta \le 0$. For $0 \le y < m$, we then have $|y - \zeta| = y - \zeta$ and $D_{98.93419}(y - m) = 98.93419(y - m)$, giving

$$g_3 = -\big[(y - \zeta) + 98.93419(y - m)\big] = -99.93419\big[y - h - \kappa_m(m - h)\big],$$
$$\kappa_m = \frac{99.93419 - 94.41370}{99.93419} \approx .0552412.$$

The third term in Equation (22) is therefore $0.59934\tanh\{99.93419[y - h - .0552412(m - h)]\}$. Its transition lies about 5.5% of the way from h to m.

The fourth term adds an asymmetric response around the market reference m, while the fifth uses a target that depends on elapsed time and order-book conditions. Both include narrow rewards for repeating the last bid, with their strength depending on the market state.

Ultimatum Game: location, concentration and lower offers. The first-stage utility combines a payoff term with asymmetric penalties around equal division. These penalties distinguish advantageous and disadvantageous inequality, as in Fehr–Schmidt preferences (Fehr & Schmidt, 1999). At stage two, GreedyBM adds a resource-dependent threshold; the predicted mean responder share rises from 28.4% to 35.9%, compared with 37.0% observed.

The third term introduces different penalties above and below a fitted reference share near one half. The stage-three model raises the probability of offers in $[.48, .52]$ from 10.2% to 28.4%, compared with 27.7% observed, with little change in the mean. Later additions reshape the remaining distribution while preserving this concentration (Figure 9). By stage five, predicted probabilities below .20 and in $[.20, .48)$ are 17.8% and 47.0%, compared with 16.9% and 47.2% observed.

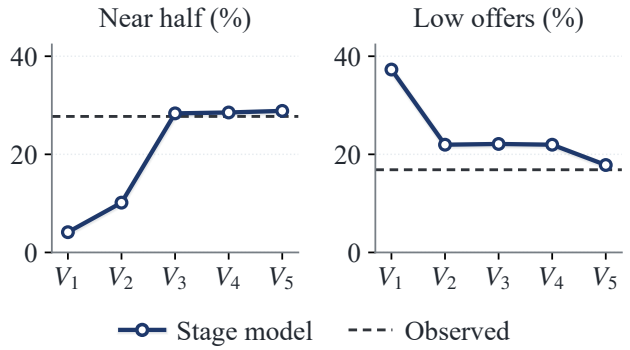


Figure 9: Ultimatum Game training behavior across stages. Dashed lines show observed offer frequencies in $[.48, .52]$ and below .20.

Let x be the resource, y the responder share, $\ell = \log(1 + x(1 - y))$, and $S(v) = \log(1 + e^v)$. With $B = 1$, the stage utility is

$$\begin{aligned}
g_1 &= \ell - ([.5 - y]_+ + \delta_0)^\alpha - \beta([y - .5]_+ + \delta_0)^\alpha,\\
g_2 &= \ell - S\{\kappa(m_0 + m_1 x^\gamma - xy)\},\\
g_3 &= \ell - S\{\phi_-(\bar{y}x - xy)\} - S\{\phi_+(xy - \bar{y}x)\},\\
g_4 &= \ell - S\{\alpha_e x[1 - 2y]_+/(x^\theta + 10^{-8})\},\\
g_5 &= \sum_{c\in\{1/2,1/3,2/3,1/4,3/4\}} w_c \exp\{-\tau(|y - c| + 10^{-12})^p\},\\
V_s(x, y) &= \sum_{j\le s} \lambda_j^{(s)} \tanh(g_j(x, y)), \qquad \delta_0 = 10^{-12}.
\end{aligned}$$

The fitted parameters, grouped by term, are

$$\begin{aligned}
g_1:&\quad \alpha = .201711, \quad \beta = 15.012509;\\
g_2:&\quad m_0 = .103759, \quad m_1 = .599290, \quad \gamma = .909793, \quad \kappa = 5.002015;\\
g_3:&\quad \bar{y} = .492749, \quad \phi_- = 6.566210, \quad \phi_+ = .252872;\\
g_4:&\quad \alpha_e = 3.868880, \quad \theta = -1.865186;\\
g_5:&\quad \tau = 21.181316, \quad p = .129155,\\
&\quad \boldsymbol{w} = (8.420790, 3.943671, .785088, 9.916907, 6.122881).
\end{aligned}$$

The entries of $\boldsymbol{w}$ follow the center order in g_5. The final outer weights are

$$\boldsymbol{\lambda}^{(5)} = (.163555, .518410, .833848, -1.815431, 1025.860245).$$

The negative fourth weight reverses the direction of its transformation. Training NLLs are 5.773960, 5.539474, 5.295679, 5.276927 and 5.050728.

C Ablation

C.1 Diversity Log versus Full Log

We evaluate how selecting diverse prompt examples affects search cost and predictive fit. Here, the diversity log is the set of representatives described in Section 3.3. We compare diversity and full logs across six paired runs per task, measuring test NLL and search tokens (Figure 10).

Replacing the full log with the diversity log reduces mean token use by 5.8% in the Ultimatum Game, 14.7% in RPS and 26.4% in CDA. With the diversity log, GreedyBM also achieves lower mean test NLL in the Ultimatum Game (5.25784 versus 5.35385) and CDA (5.08160 versus 5.10123). In RPS, mean test NLL is 1.08275 with the diversity log and 1.08259 with the full log.

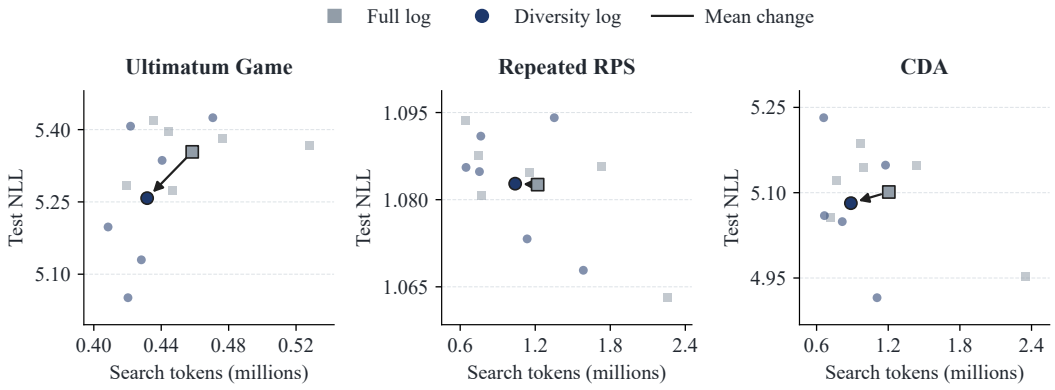


Figure 10: Diversity-log ablation: test NLL versus search tokens. Squares denote the full log; circles denote the diversity log. Faint markers show six paired runs per task, outlined markers show means, and arrows connect full-log means to diversity-log means.

C.2 Candidate Age Replay

To assess the need to generate new candidates at every stage, we compare candidates from earlier stages with fresh candidates proposed at stage t, all evaluated at the current utility V_{t-1}. In *reoptimized replay*, we tune an earlier candidate's term parameters $\widehat{\theta}$ to maximize the squared score at fixed V_{t-1}; *frozen replay* retains its source-stage parameters. Both modes then minimize training NLL over all t weights $\widehat{\boldsymbol{\lambda}}$ with term parameters fixed.

Let $\mathcal{R}_{s,t}$ contain stage-s candidates with valid replay fits at stage t. Their largest training-NLL gain is

$$G(s,t) = \max_{f \in \mathcal{R}_{s,t}} \left\{ \widehat{L}_G(V_{t-1}) - \widehat{L}_G\left(\sum_{j<t} \widehat{\lambda}_j f_j + \widehat{\lambda}_t f(\cdot; \widehat{\theta}) \right) \right\}.$$

Positive $D(s,t) = G(t,t) - G(s,t)$ favors fresh candidates.

We evaluate all ten stage pairs at exploration probabilities 0.2–0.6, averaging five runs per cell. Fresh candidates yield larger mean gains in 26/50 Ultimatum Game, 38/50 RPS and 43/50 CDA cells with reoptimized replay; the frozen-replay counts are 29/50, 37/50 and 44/50. Retuning preserves the advantage of fresh candidates in most RPS and CDA cells; Ultimatum Game cells split roughly evenly.

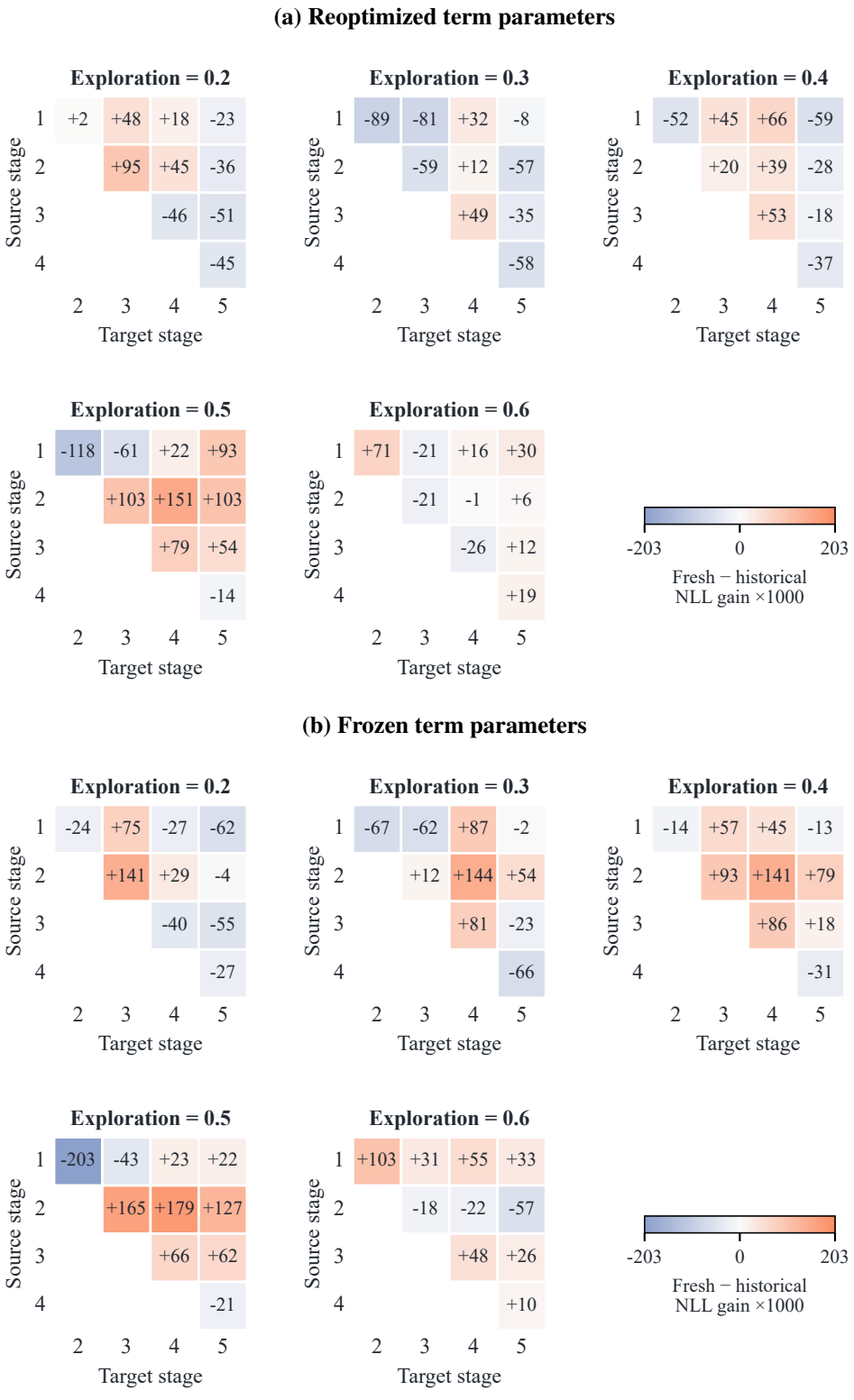


Figure 11: Candidate age replay in the Ultimatum Game with (a) reoptimized and (b) frozen term parameters. Cells show five-run mean $D(s,t) \times 1000$. Blue favors earlier candidates; orange favors fresh candidates. Within each task, the color scale is shared across modes and exploration settings.

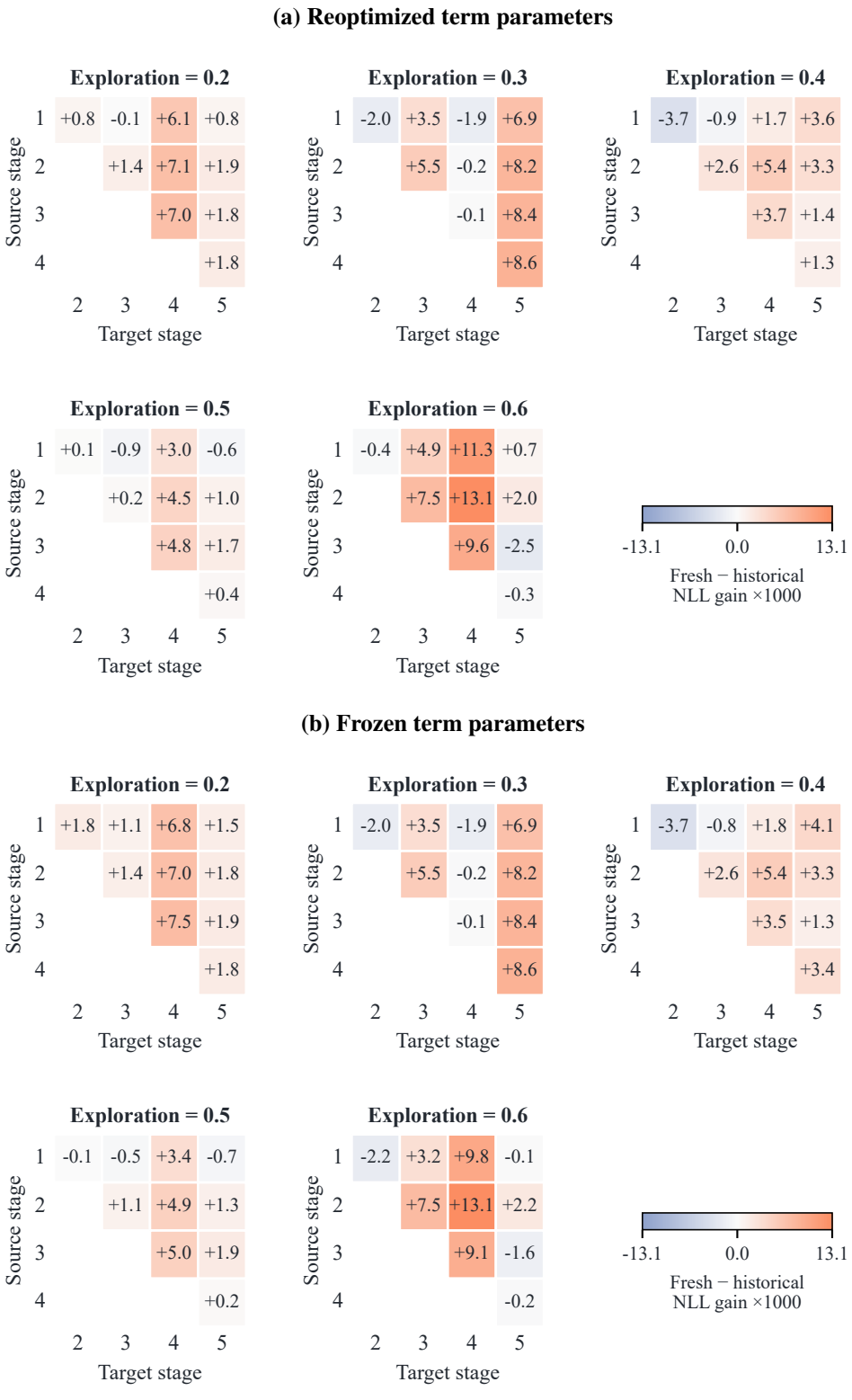


Figure 12: Candidate age replay in RPS with (a) reoptimized and (b) frozen term parameters. Cells show five-run mean $D(s,t) \times 1000$. Blue favors earlier candidates; orange favors fresh candidates. Within each task, the color scale is shared across modes and exploration settings.

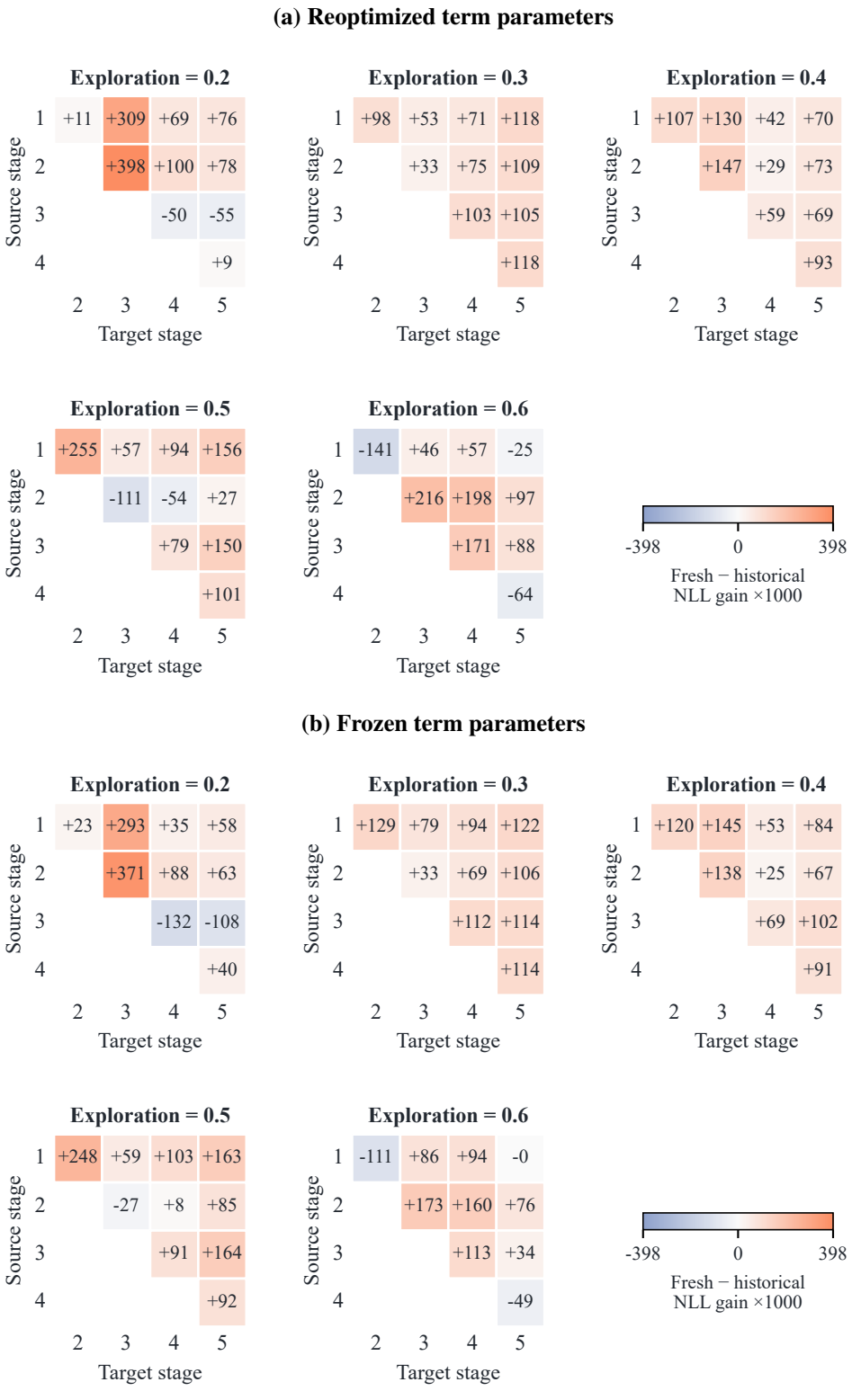


Figure 13: Candidate age replay in CDA with (a) reoptimized and (b) frozen term parameters. Cells show five-run mean $D(s,t) \times 1000$. Blue favors earlier candidates; orange favors fresh candidates. Within each task, the color scale is shared across modes and exploration settings.