APS: Bias-Controlled Adaptive Prototype Simulation for Population-Scale LLM Agents

Quan Zheng[1,2], **Yan Gao**[2,3]*, **Shaobin He**[2,4], **Haoxiang Guan**[2,5]
Jie Feng[2,3], **Shuxin Zheng**[2,3], **Yuanhe Tian**[2,3], **Ming Wang**[1], **Zhen Liu**[2]
[1]Beijing Normal University [2]Zhongguancun Academy, Beijing, China
[3]Zhongguancun Institute of Artificial Intelligence
[4]Tianjin University [5]University of Science and Technology of China
s-zq25@bza.edu.cn, gaoyan@zgci.ac.cn

Abstract

LLM-agent simulation offers a flexible computational tool for studying population response trajectories. However, full multi-round simulation scales linearly with both population size and horizon, requiring every agent to query the LLM at every round. We propose **Adaptive Prototype Simulation (APS)**, a framework that treats scaling as a recurrent LLM-oracle allocation problem. APS preserves the specified LLM as the online transition oracle while querying adaptive core prototypes, selected singleton-tail agents, and shadow-audit agents. Prototype responses induce local response surfaces for nearby agents, reducing online LLM calls without replacing the underlying transition model. To control approximation bias, shadow-audit residual correction estimates propagation residuals for aggregate correction and future budget allocation, while tail-protected singleton routing directly queries selected isolated, heterogeneous, or high-curvature regions that are vulnerable to smoothing. Under the reported protocols, APS achieves lower distributional discrepancy in simulation results while reducing online LLM calls. In a 10M-agent, multi-round public-opinion simulation, APS achieves a $381.1\times$ reduction over full simulation, with reference-aligned final-round JSD 0.094 against the corresponding full-LLM reference.[1]

1 Introduction

Large language models make it possible to build social agents whose responses depend on scenario prompts, memory, demographics, and evolving social context [Park et al., 2023, Wang et al., 2024, Guo et al., 2024]. This can expand classical agent-based modeling beyond hand-crafted transition rules [Epstein and Axtell, 1996, Macy and Willer, 2002, Castellano et al., 2009], but it creates a computational bottleneck: simulating N agents for T rounds requires $O(NT)$ online LLM calls.

Existing strategies either keep rich prompts but simulate only small "micro-societies" [Park et al., 2023, Li et al., 2023], or reduce repeated LLM calls through serving optimizations, routing, or learned proxies [Frantar et al., 2023, Leviathan et al., 2023, Chen et al., 2023a, Kwon et al., 2023, Guan et al., 2025]. These strategies alone do not provide a recurrent population-transition estimator that keeps the specified LLM online while accounting for propagation bias and tail smoothing.

We propose **Adaptive Prototype Simulation (APS)**, a framework for scalable LLM-agent population simulation that treats scaling as recurrent LLM-oracle allocation. APS selects representative prototypes, queries their LLM-induced responses under the current scenario and social context, and applies these transitions to similar agents. Across rounds, APS reallocates prototype budgets using stratum

*Corresponding author.
[1]Method reference implementation: https://github.com/zgcai-abss/APS.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

size, residual error, and local similarity and curvature diagnostics rather than a fixed representative schedule. Compared with prior cost-reduction strategies for large-scale simulation, APS reduces online calls without replacing the specified LLM transition with a trained proxy; queried prototype LLM responses remain the semantic anchors for same-round propagation.

The key challenge is approximation bias. Agents assigned to similar prototypes can still exhibit different LLM-induced transitions, and this mismatch can affect later prompts through recurrent states and neighbor summaries. APS uses two complementary bias-control mechanisms. Shadow-audit residual correction uses shadow-audit queries to estimate discrepancies between local-response-surface predictions and individual LLM responses; these residuals correct aggregate transition estimates and inform future budget allocation toward high-error, high-curvature, or low-recall regions. Tail-protected singleton routing directly queries selected isolated feature-space agents and removes them from the smoothing operator, reducing the risk that selected isolated, heterogeneous, or high-curvature feature-space regions are absorbed by dominant prototypes.

We study the computational problem of approximating the population transition induced by full LLM-agent simulation, not the behavioral-validity problem of proving that LLMs model human populations. The central idea is to turn prototype propagation from an unchecked compression heuristic into an audited oracle-allocation loop: prototypes anchor same-round semantics, tail-protected singleton routing reduces smoothing of selected isolated agents, and shadow-audit residuals correct aggregate estimates while steering future queries. APS is specified by oracle anchoring, tail-protected singleton routing, shadow-audit residual correction, and the separation of recurrent hard states from audit-corrected reporting; the clustering method and prototype-response interpolation rule are replaceable implementation choices, and our empirical claims are limited to the reported protocols.

Our contributions are organized in three layers:

- **APS framework for population-scale LLM agents.** APS keeps the LLM as the online transition oracle, queries adaptive core prototypes each round, and propagates their same-round transitions to the full population while maintaining recurrent social context. Its allocation rule dynamically shifts prototype budgets across strata according to residual error, local similarity diagnostics, and stratum size, rather than fixed representative quotas.
- **Bias control for prototype propagation.** APS reduces approximation bias introduced by prototype propagation through shadow-audit residual correction and tail-protected singleton routing. Shadow-audit residuals correct aggregate estimates and inform future budget allocation toward high-error, high-curvature, or low-recall regions; tail-protected singleton routing directly queries selected isolated agents vulnerable to prototype smoothing.
- **Analysis and large-scale evaluation.** We formalize the dynamic LLM transition estimand and analyze prototype coverage, audit correction, local propagation, and temporal context mismatch. Evaluation covers a 10M-agent, 8-round simulation with $381.1\times$ fewer calls, same-budget and strengthened baselines, component and allocation ablations, and runtime. Further tests examine 50-round stability, context sensitivity, grouping and budget–fidelity trade-offs, robustness across scenarios and backends, cross-task performance, and dynamic-memory extensions.

2 Related Work

Generative agents and related systems show that LLMs can drive memory-based routines, social interaction, role play, and survey-like behavior in small-to-medium populations [Park et al., 2023, Li et al., 2023, Chen et al., 2023b, Zhou et al., 2023, Argyle et al., 2023, Aher et al., 2023, Park et al., 2024, Ashery et al., 2025]. As researchers use LLM agents to study population-level regularities closer to real-world social dynamics, recent surveys document this shift, and large-scale systems and the large population models framework scale simulation via platforms, learned surrogates, differentiable population-scale frameworks, and topology-aware grouping [Gao et al., 2024, Mou et al., 2024, Gao et al., 2023, Vezhnevets et al., 2023, Tang et al., 2025, Piao et al., 2025, Guan et al., 2025, Chopra, 2025, Xu et al., 2026]. This shift makes cost a central bottleneck, and serving work lowers per-call inference cost through quantization, speculative decoding, routing, and memory management [Frantar et al., 2023, Leviathan et al., 2023, Chen et al., 2023a, Sheng et al., 2023, Kwon et al., 2023]. However, cost-reduction strategies often trade off LLM anchoring or error control: learned proxies move the transition rule into trained models, grouped representatives or static propagation leave

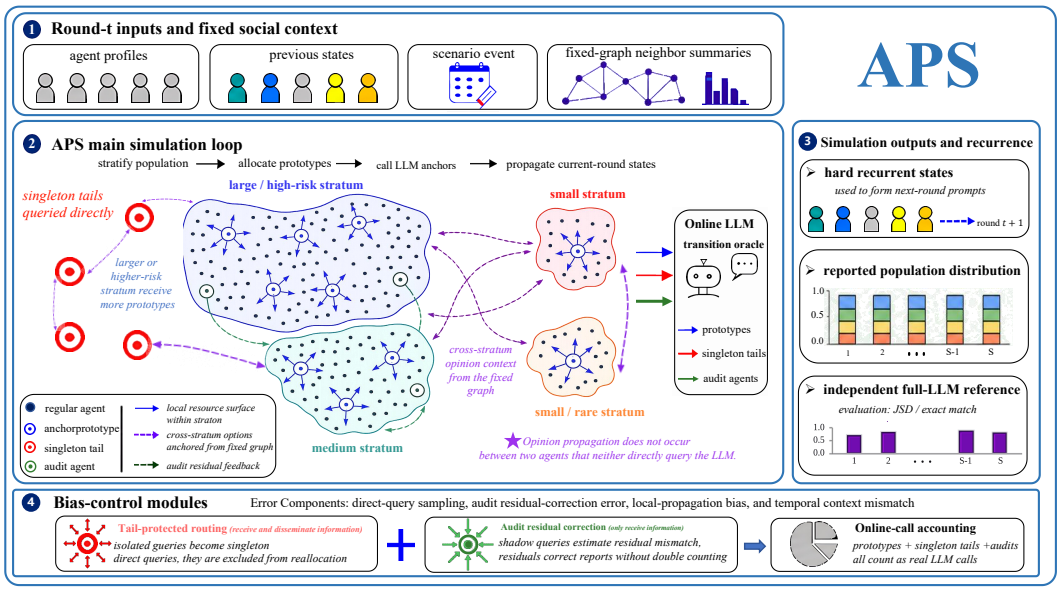


Figure 1. Overview of Adaptive Prototype Simulation (APS). Each round, APS builds prompts from profiles, scenario events, pre-round hard recurrent states, and fixed-graph summaries. It queries budgeted core-prototype, singleton-tail, and shadow-audit agents, then propagates prototype responses within core strata. Tail-protected singleton routing and shadow-audit residual correction are the two bias-control modules. Total LLM calls comprise core-prototype, singleton-tail, and shadow-audit queries.

unqueried agents dependent on local approximations, and serving-level optimizations reduce per-call cost without defining a recurrent population-transition estimator with propagation-bias diagnostics.

Compared with these lines of work, APS addresses the cost–fidelity gap by keeping the specified LLM as the online transition oracle while reducing online calls through adaptive prototype simulation. It queries core prototypes, singleton-tail agents, and a small shadow-audit sample, propagates same-round prototype responses through local response surfaces, and dynamically allocates prototype budgets using stratum size, residual error, and local similarity diagnostics. For error control, shadow-audit residual correction estimates propagation residuals for aggregate correction and future budget allocation, while tail-protected singleton routing directly queries selected isolated, heterogeneous, or high-curvature feature-space regions vulnerable to smoothing. Additional related-work details are discussed in Appendix A.

3 Adaptive Prototype Simulation

APS treats the specified LLM, prompt, scenario, option set, parser, and decoding rule as the online transition oracle, but queries only a selected subset of agents each round. The method section is organized as follows. Section 3.1 defines the problem setting, computational targets, and state records APS maintains; Section 3.2 gives an overview of the APS workflow, including prototype propagation and query allocation; Section 3.3 explains the two bias-control mechanisms, tail-protected singleton routing and shadow-audit residual correction; and Section 3.4 summarizes the resulting query complexity and error decomposition. Operational method details are collected in Appendix B, and derivations are moved to Appendix C so that the main text emphasizes the simulation logic.

3.1 Problem Definition, Targets, and State Records

We consider a population $\mathcal{A}_N = \{a_i\}_{i=1}^N$ of N agents and a finite decision set $\mathcal{Y}$. Agent i has static profile features x_i and a position in a fixed social graph G. At round t, the scenario event is s_t, and the prompt for an agent includes its profile, its previous decision, and a summary of previous-round neighbor decisions. We write $K_\theta(y \mid z, s_t)$ for the decision distribution induced by the fixed LLM pipeline under prompt context z; here θ denotes the fixed model, prompt, parser, and decoding configuration. Deterministic decoding is the point-mass special case.

A full-LLM reference simulation, abbreviated as the full-reference simulation, queries every agent at every round. Its context for agent i is $z_i^{\star,t}$, built from the agent profile, the full-reference previous state, and the corresponding neighbor summary, and its realized decision is $y_i^{\star,t}$. The expected full-reference population transition $\bar{p}_t^\star$ and the realized empirical reference distribution $p_t^\star$ are

$$\bar{p}_t^\star(y) = N^{-1} \sum_i K_\theta(y \mid z_i^{\star,t}, s_t), \qquad p_t^\star(y) = N^{-1} \sum_i \mathbb{I}[y_i^{\star,t} = y]. \tag{1}$$

APS cannot query every $z_i^{\star,t}$. Instead, it maintains an approximate hard recurrent state $\hat{y}_i^t$ for each agent and builds APS-context prompts $\hat{z}_i^t$ from these states. If every agent were queried under these APS-maintained contexts, the target would be $\bar{p}_t^A(y) = N^{-1} \sum_i K_\theta(y \mid \hat{z}_i^t, s_t)$. The gap between $\bar{p}_t^A$ and $\bar{p}_t^\star$ is the temporal context mismatch from approximate recurrence. In the API experiments, we observe realized categorical decisions, not calibrated option probabilities, so reference-aligned metrics compare against the empirical distribution $p_t^\star$. The full-reference simulation is a computational target for the specified LLM system, not a claim about true human behavior.

APS therefore keeps two role-specific records. The hard recurrent ledger $\hat{y}_i^t$ advances future prompts and neighbor summaries through APS-maintained contexts. The soft reporting ledger h_i^t stores a per-agent, per-category probability vector over $\mathcal{Y}$: one-hot LLM decisions for directly queried agents and interpolated response-surface probabilities for propagated agents. The reported population distribution is the simplex-projected audit-corrected aggregate built from the soft ledger and design-weighted shadow-audit residuals, not the uncorrected empirical count of hard recurrent states.

3.2 APS Workflow Overview

Figure 1 shows the APS workflow. Before the first round, APS selects a singleton-tail set O_N for tail-protected singleton routing and partitions the remaining agents into core strata $\{C_m\}$.[2] At round t, it builds prompts from the pre-round hard recurrent states, chooses prototype sets $S_{m,t} \subset C_m$, queries the LLM for those prototypes and the singleton-tail agents, and propagates prototype decisions to non-prototype core agents through a local response surface. The online calls in a round thus consist of core-prototype queries, singleton-tail queries, and the shadow-audit queries defined below; only the first two update recurrent hard states. Detailed step-by-step pseudocode is given in Appendix B.1.

For a direct rollout agent, namely a core prototype or singleton-tail agent, the soft vector h_i^t is the one-hot encoding of the parsed LLM decision. For a non-prototype core agent $j \in C_m \setminus S_{m,t}$, let $P_\kappa(j,t) \subseteq S_{m,t}$ be the set of up to κ nearest queried prototypes within the same stratum, and let $w_{ji,t}$ be nonnegative interpolation weights with $\sum_{i \in P_\kappa(j,t)} w_{ji,t} = 1$. APS sets

$$h_j^t(y) = \sum_{i \in P_\kappa(j,t)} w_{ji,t}\, \mathbb{I}[\hat{y}_i^t = y], \qquad \hat{y}_j^t = \arg\max_{y \in \mathcal{Y}} h_j^t(y). \tag{2}$$

Ties are broken by a fixed rule. This separation lets APS use hard states for recurrence while keeping soft local-response-surface information for aggregate reporting and shadow-audit residual correction.

The query budget is controlled by sublinear scale schedules. Core strata grow as $M_{\text{core}}(N) = O(N^\eta)$ and the singleton-tail set grows as $M_{\text{out}}(N) = O(N^\zeta)$, with sublinear exponents $\eta, \zeta < 1$, so absolute counts grow while fractions shrink. The core prototype rate follows

$$\alpha(N) = \begin{cases} \alpha_b, & N \le N_b, \\ \alpha_b (N_b/N)^\lambda, & N > N_b, \end{cases} \qquad 0 < \lambda < 1. \tag{3}$$

Here N_b is the baseline scale, α_b the baseline prototype rate, and λ the decay exponent. For $N > N_b$, the queried fraction decreases but the absolute number of core prototypes grows.

If $B_t^{\text{core}} = \lceil \alpha(N) N \rceil$ denotes the nominal pre-round core-prototype budget and $R_{m,t-1}$ is the previous residual-risk score defined in Section 3.3 with $R_{m,0} = 1$, the continuous allocation for stratum C_m is

$$\widetilde{B}_{m,t} = B_t^{\text{core}} \frac{|C_m| \sqrt{R_{m,t-1} + \tau}}{\sum_\ell |C_\ell| \sqrt{R_{\ell,t-1} + \tau}}, \tag{4}$$

[2] In the main experiments, core strata are fixed because agents' profile features are treated as static. In settings where profile features evolve, APS should include an explicit profile-transition mechanism and recompute feature-space tail scores and core strata every round or after a fixed number of rounds. A pilot experiment is reported in Appendix B.5.

where the denominator sums over all core strata to normalize allocation weights and $\tau > 0$ stabilizes strata with near-zero scores. The rollout obtains the integer budget $B_{m,t}$ by rounding $\widetilde{B}_{m,t}$ and applying the minimum-one rule for nonempty strata described in Appendix B.3.

3.3 Bias-Controlled Routing and Auditing

Prototype propagation is efficient only if local similarity is informative. APS therefore adds two safeguards that target distinct smoothing and residual-error failure modes.

Tail-protected singleton routing. Local smoothing can distort isolated or heterogeneous feature regions when no close prototype is queried. APS therefore uses a feature-space tail score s_i for each agent and directly queries the top-scoring $M_{\text{out}}(N)$ agents every round. Here s_i is a robust standardized distance from the feature median; Appendix D.1.3 gives the exact scoring rule. These singleton-tail agents contribute one-hot soft vectors for themselves, but they are not used as prototype supports for other agents during local interpolation within core response surfaces. This reduces smoothing risk for selected isolated regions; it does not claim to identify all minority opinions.

Shadow-audit residual correction. After propagation, APS audits a subset of non-prototype core agents. These shadow-audit queries use the same LLM, prompt format, scenario event, and APS-maintained context as rollout calls, but their labels are shadow observations: they do not enter the prototype set and do not overwrite current hard states. Let U_t be the audit set, $\tilde{y}_i^t$ the shadow-audit label for audited agent i, and $\psi_{i,t}$ the known probability that this agent is included in the shadow-audit sample. APS reports

$$\widehat{p}_t^{\text{audit}}(y) = \frac{1}{N}\sum_{i=1}^{N} h_i^t(y) + \frac{1}{N}\sum_{i\in U_t} \frac{\mathbb{I}[\tilde{y}_i^t = y] - h_i^t(y)}{\psi_{i,t}}. \tag{5}$$

The first term is the full-population local-response-surface baseline; the second is a design-weighted residual correction from shadow-audit queries. For JSD reporting, APS clips and renormalizes it to the probability simplex. With positive shadow-audit inclusion probabilities, the local response surface need not be correct for the unprojected estimator to be design-valid for the APS-context target; poor propagation instead appears as larger residual variance and later shadow-audit demand. The shadow-audit sampling rule and allocation diagnostics are given in Appendices B.1 and B.2.

Shadow-audit queries also update the next-round residual-risk score used in Eq. 4. For stratum C_m, APS combines normalized shadow-audit diagnostics as

$$R_{m,t} = \widehat{V}_{m,t}^{\text{res}} + \lambda_\rho \widehat{L}_{m,t}^2 \widehat{\rho}_{m,t}^2 + \lambda_e \widehat{e}_{m,t}^2 + \lambda_r (1 - \widehat{r}_{m,t})^2. \tag{6}$$

Here $\widehat{V}_{m,t}^{\text{res}}$ is residual variance, $\widehat{\rho}_{m,t}$ is mean distance from audited agents to prototype support, and $\widehat{L}_{m,t}$ is local response disagreement or curvature. $\widehat{e}_{m,t}$ measures hard-state mismatch with audit labels, and $\widehat{r}_{m,t}$ is rare-state recall. These diagnostics guide later allocation.[3] The weights $\lambda_\rho, \lambda_e, \lambda_r$ control their relative contributions.

3.4 Complexity and Error Analysis

The analysis concerns computational fidelity to the LLM-induced population transition, not its behavioral validity for real human populations. Counting core-prototype queries, singleton-tail queries, and shadow-audit queries, APS reduces the $O(TN)$ calls of full simulation to

$$C_{\text{APS}}(N,T) = O\big(T\left[N^{1-\lambda} + M_{\text{core}}(N) + M_{\text{out}}(N) + A(N)\right]\big) \tag{7}$$

Here T counts rounds. Per round, $N^{1-\lambda}$ covers the nominal prototype budget, and $M_{\text{core}}(N)$ covers the per-stratum minimum. $M_{\text{out}}(N)$ counts singleton-tail queries, and $A(N)$ is the order of shadow-audit calls $A_t(N)$. Under $A(N) = O(N^{1-\lambda})$, total calls are $O(T[N^{1-\lambda} + M_{\text{tot}}(N)])$, with $M_{\text{tot}}(N) = M_{\text{core}}(N) + M_{\text{out}}(N)$, sublinear in N for schedule exponents below one.

The error decomposition tracks the four sources used across the paper: prototype-coverage error $\epsilon_{\text{cov},t}$, finite shadow-audit residual-correction error $\epsilon_{\text{audit},t}$, core local-propagation bias $\epsilon_{\text{prop},t}$, and

[3] Appendix B.2 defines the residual, mismatch, support, and rare-response diagnostics.

temporal context mismatch $\epsilon_{\mathrm{ctx},t}$. The first three form the current-round APS-context error; the fourth arises as hard recurrent states condition future prompts. If $e_t^\star$ is the induced full-reference error, then

$$e_t^\star \le \epsilon_{\mathrm{cov},t} + \epsilon_{\mathrm{audit},t} + \epsilon_{\mathrm{prop},t} + \epsilon_{\mathrm{ctx},t}. \tag{8}$$

Shadow-audit residual correction targets local-propagation bias in the unprojected aggregate estimator, rather than directly repairing the propagated hard states themselves, but poor propagation can still raise finite-audit variance and affect later rounds through hard recurrent states. Appendix C gives the formal bounds, including design validity, residual-variance comparison, tail-bias removal, local-response-surface regularity, multi-round recurrence, and residual-aware allocation.

4 Experiments and Results

This section first describes the experimental setup and then addresses the following research questions:

- **RQ1:** How accurately and efficiently does APS match references at 10M-agent scale?
- **RQ2:** Under matched budgets, does APS outperform approximation baselines?
- **RQ3:** What do ablations show about component gains and cost?
- **RQ4:** Does APS avoid sustained error accumulation across rounds?
- **RQ5:** How do grouping and query budgets affect fidelity, and how robust is APS across settings?

4.1 Experimental Setup

Scenario and populations. We evaluate APS on an 8-round public-opinion simulation about a hypothetical subway chemical attack. Each round provides new information and asks agents to choose among five options; the task tests macro-level opinion dynamics rather than crisis policy. Agent profiles start from about 90K real respondents in the multi-country World Values Survey (WVS) [Haerpfer et al., 2022], represented with 19 standardized survey-derived features. These profiles are simulation inputs, not city-population claims. Larger populations use type-aware, stratum-preserving perturbations around real WVS records; Appendix D.1.1 gives the construction protocol.

Methods and operating point. We evaluate APS against independent full-LLM reference rollouts and compare it with LS-Surrogate adapted from Light Society [Guan et al., 2025] and TopoSim-Coord adapted from TopoSim [Xu et al., 2026]. Unless otherwise stated, APS uses $N_b = 5000$, $\alpha_b = 0.15$, baseline core-stratum count $M_b = 10$, $\lambda = 0.6$, $\eta = 0.5$, $\zeta = 0.4$, and $\beta_a = 1 - \lambda = 0.4$; Appendix B.4 lists the Table 1 superscript settings. The 3K–100K rows use OpenRouter with glm-4.7-flash under a one-process four-key configuration. The 1M and 10M rows use the official Zhipu API with glm-4-flash; this provider split is an API-cost constraint, as the large-scale runs use the no-cost official Zhipu endpoint available to us. All GLM runs use deterministic decoding; Appendix D gives API settings, runtime and cost accounting, and Appendix F gives prompt text.

Independent full-LLM reference rollouts. Full-LLM reference rollouts are independent of APS: they update the corresponding population through all rounds with their own recurrent states and neighbor summaries, and no APS state, propagated label, audit result, or budget decision enters the reference. These rollouts are computational targets, not evidence of human behavioral validity; protocol and accounting details are in Appendices D.3.1–D.4.2.

Metrics. We measure distributional, agent-level, and graph-context fidelity against Full-LLM references, alongside LLM-call cost. Jensen–Shannon divergence (JSD) compares reported distributions with Full-LLM references, using the audit-corrected distribution for APS. Hard-ledger JSD compares recurrent hard-state distributions. Exact match measures agreement between aligned recurrent hard top-1 states, providing an auxiliary agent-level diagnostic. Mean neighborhood total variation (TV) averages, over agents, half the summed absolute differences between APS and Full-LLM neighbor-state probabilities. TV ranges from 0 for identical distributions to 1 for maximally different distributions. Edge joint-state JSD compares state-pair distributions of connected agents. Confidence-interval procedures and long-horizon metric definitions and results appear in Appendices D.3.2 and E.2.1.

4.2 Final-Round Accuracy, Cost, and 10M Scaling (RQ1)

Table 1 is the main provider-aware scale comparison against LS-Surrogate [Guan et al., 2025] and TopoSim-Coord [Xu et al., 2026].[4] Reference outputs are used for JSD and exact-match

[4] Appendix E.1 evaluates strengthened variants under matched budgets, adding APS-aligned recurrent inputs and shared CommonAudit correction beyond the baselines' native designs.

Table 1. Final-round JSD, exact-match accuracy, and scaling. Superscripts on APS denote small-scale validation (†), high-budget validation (‡), and production schedule (*). Reduction is the Full-LLM-to-method call ratio. JSD and exact match use the corresponding Full-LLM reference. Brackets report 95% intervals.

Scale	Method	LLM Calls	Reduction	JSD ↓	JSD CI	Exact ↑	Exact CI
3K	APS†	8.9K	**2.7**×	**0.011**	[0.0065, 0.0155]	**0.673**	[0.6560, 0.6896]
10K	LS-Surrogate	8.0K	**10.0**×	0.166	[0.1570, 0.1750]	0.490	[0.4802, 0.4998]
10K	TopoSim-Coord	20.7K	3.9×	0.150	[0.1414, 0.1586]	0.344	[0.3348, 0.3534]
10K	APS‡	20.8K	3.8×	**0.014**	[0.0112, 0.0268]	**0.662**	[0.6527, 0.6712]
100K	LS-Surrogate	80.0K	10.0×	0.179	[0.1761, 0.1819]	0.486	[0.4829, 0.4891]
100K	TopoSim-Coord	52.2K	**15.3**×	0.153	[0.1503, 0.1557]	0.338	[0.3351, 0.3409]
100K	APS‡	172.0K	4.7×	**0.021**	[0.0199, 0.0221]	**0.639**	[0.6360, 0.6420]
1M	LS-Surrogate	800.0K	10.0×	0.223	[0.2220, 0.2240]	0.450	[0.4490, 0.4510]
1M	TopoSim-Coord	130.2K	61.4×	0.172	[0.1711, 0.1729]	0.334	[0.3331, 0.3349]
1M	APS*	83.4K	**95.9**×	**0.062**	[0.0614, 0.0626]	**0.517**	[0.5160, 0.5180]
10M	LS-Surrogate	8.0M	10.0×	0.264	[0.2637, 0.2643]	0.411	[0.4107, 0.4113]
10M	TopoSim-Coord	329.7K	242.6×	0.191	[0.1907, 0.1913]	0.325	[0.3247, 0.3253]
10M	APS*	209.9K	**381.1**×	**0.094**	[0.0938, 0.0942]	**0.544**	[0.5437, 0.5443]

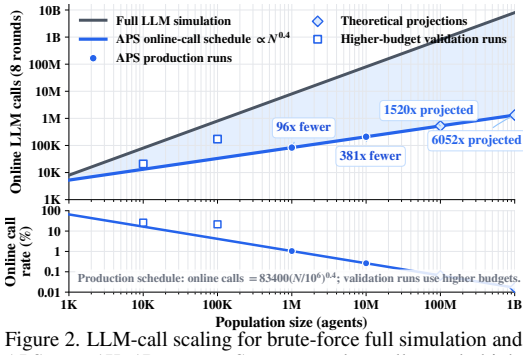


Figure 2. LLM-call scaling for brute-force full simulation and APS over 1K–1B agents. Squares mark smaller-scale high-budget APS validation results, filled circles mark completed production runs up to 10M, and diamonds show theoretical 100M and 1B schedule projections.

Method	Calls	JSD	Exact
Strat. sample [Wu and Thompson, 2020]	15.5K	0.088	0.515
Cluster assign., App. D.2.4	15.5K	0.167	0.321
Label prop. [Xing et al., 2022]	15.5K	0.061	0.513
Core-set [Feldman, 2020]	15.5K	0.124	0.408
Strat. emp. [Wu and Thompson, 2020]	15.5K	0.016	0.539
Active sel. [Diao et al., 2024]	15.5K	0.033	0.514
Centroid reps. [Bemporad, 2023]	15.5K	0.029	0.605
Online surr. [Diaw et al., 2024]	15.5K	0.017	0.589
APS	15.5K	**0.011**	**0.624**

Table 2. Same-budget 10K approximation baselines, counting all online calls. JSD measures aggregate error; exact match compares recurrent hard states with Full. References identify method families; the task-aligned implementations used in this comparison are specified in Appendix D.2.4.

reporting. LS-Surrogate counts reference-derived distillation labels, while TopoSim-Coord counts online representative queries. APS counts core-prototype, singleton-tail, and shadow-audit queries, without using reference labels for propagation or adaptation. Reduction divides Full-LLM calls by each method's counted calls. Appendix D gives implementation, reference, call-accounting, runtime, and cost details.

The accuracy trend is consistent across matched scales: where baselines are reported, APS has the lowest final-round JSD and the highest exact-match diagnostic. At 10K agents, for example, APS reduces JSD from 0.166 for LS-Surrogate and 0.150 for TopoSim-Coord to 0.014. At 10M agents under the production schedule, APS obtains JSD 0.094 with 95% CI [0.0938, 0.0942], compared with 0.264 for LS-Surrogate and 0.191 for TopoSim-Coord.

Figure 2 separates the cost trend. At 10M agents over eight rounds, brute-force full simulation requires 80M LLM calls, whereas APS uses 209.9K counted online calls, including core, singleton-tail, and audit queries, a 381.1× reduction over brute force; under the same Zhipu configuration, wall-clock time is 26.64 hours for APS[5] and 9.56 days for the full-LLM reference rollout. Across the reported scales, APS achieves lower JSD and higher exact match than both scale-oriented baselines. Relative to full-LLM simulation, it reduces both LLM calls and end-to-end runtime.

4.3 Same-Budget Approximation Baselines (RQ2)

Table 2 fixes the 10K scale and 15.5K online-call budget over eight rounds, then compares APS with sampling, propagation, selection, coreset, and surrogate approximations under the same online-call

[5]For APS, wall-clock time is dominated by prototype clustering and simulation-pipeline processing, including feature processing, propagation, audit aggregation, checkpointing, and bookkeeping; API waiting is a smaller share under this multi-key setup.

Table 3. 10K ablations for shadow-audit residual correction and tail-protected singleton routing.

Variant	LLM Calls	JSD ↓	JSD 95% CI	Exact match ↑	Exact match 95% CI
Shadow-audit residual correction					
Base APS without residual correction	9.3K	0.098	[0.090, 0.106]	0.536	[0.526, 0.546]
APS without residual correction (same budget)	12.3K	0.075	[0.066, 0.082]	0.584	[0.573, 0.591]
APS with residual correction	12.3K	0.019	[0.014, 0.024]	0.659	[0.650, 0.668]
Tail-protected singleton routing					
APS without singleton-tail routing	9.4K	0.030	[0.022, 0.038]	0.529	[0.519, 0.539]
APS without singleton-tail routing (same budget)	12.3K	0.022	[0.017, 0.028]	0.591	[0.584, 0.599]
APS with singleton-tail routing	12.3K	0.011	[0.007, 0.019]	0.662	[0.654, 0.671]

Rate	Allocation	Seeds	Calls	JSD	Exact
0.05	fixed	2	5.3K	0.064	0.373
0.05	adaptive	2	5.3K	0.021	0.489
0.10	fixed	2	6.5K	0.056	0.392
0.10	adaptive	2	6.5K	0.018	0.506
0.15	fixed	2	7.7K	0.052	0.424
0.15	adaptive	2	7.7K	0.020	0.542
0.20	fixed	2	8.9K	0.047	0.457
0.20	adaptive	2	8.9K	0.018	0.625
0.30	fixed	2	11.3K	0.042	0.506
0.30	adaptive	2	11.3K	0.011	0.638
0.50	fixed	2	16.2K	0.026	0.507
0.50	adaptive	2	16.2K	0.006	0.664

Table 4. Adaptive allocation ablation at 3K scale, averaged over two seeds. Calls include core-prototype, singleton-tail, and shadow-audit queries across all eight rounds.

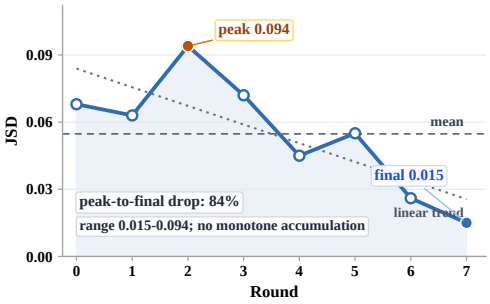


Figure 3. Per-round JSD in the completed 10K full-LLM trajectory check, with mean and linear-trend diagnostics; the nonmonotone path is consistent with no monotone accumulation in this check.

budget, population, scenario, parser, LLM endpoint, and reference task, isolating allocation and propagation choices rather than scale, prompt, or reference effects. APS has the lowest final JSD, the primary distributional metric, and highest exact match, an auxiliary hard-state diagnostic, in this controlled comparison; Appendix D.2.4 gives implementation details.

4.4 Component and Allocation Ablations with Complexity Trade-offs (RQ3)

Table 3 reports 10K component checks for the two bias-control modules. At matched 12.3K calls, the two modules reduce JSD from 0.075 to 0.019 and 0.022 to 0.011, respectively, while raising exact match from 0.584 to 0.659 and 0.591 to 0.662. The lower-call component-off settings show the same direction, with larger errors when either module is disabled; Appendix D.3.3 gives protocol details.

Table 4 compares adaptive and fixed allocation at matched 3K budgets, counting prototype, singleton-tail, and audit calls. This isolates allocation from total-query effects: adaptive allocation lowers JSD at every rate and reaches exact match 0.664; fixed allocation remains at or below 0.507.

Component-off controls spend the same calls elsewhere. The components therefore improve fidelity without increasing the matched LLM-call budget. Appendix D.4.2 reports APS's end-to-end runtime advantage after accounting for all component overhead. Appendix E.3.2 quantifies the cost–fidelity trade-off across query budgets using corrected JSD and exact match.

4.5 Multi-Round Stability and Error Accumulation (RQ4)

We compare APS with an independent Full-LLM rollout on the same 10K agents over eight rounds under the same scenario stages. Figure 3 shows a nonmonotone JSD path that peaks at 0.094 and ends at 0.015. We did not observe monotone error growth over these eight rounds.

Table 5. Paired 10K-agent, 50-round fidelity. Mean and worst cover all rounds. Worst is the minimum exact match or maximum error.

Metric	Mean	Worst	R30	R50
Audit-corrected JSD ↓	0.037	0.089	0.085	0.018
Hard-ledger JSD ↓	0.052	0.116	0.112	0.031
Hard exact match ↑	0.608	0.530	0.542	0.635
Mean neighborhood TV ↓	0.228	0.347	0.341	0.158
Edge joint-state JSD ↓	0.055	0.111	0.106	0.033

Table 5 reports recurrent-state and graph-context fidelity for a paired 10K-agent, 50-round trajectory. All five reported fidelity measures improve from R30 to R50. Trends in corrected JSD, hard exact match, and neighborhood TV over R11–50 agree with their early/late window means (R11–20 versus R41–50). This paired trajectory

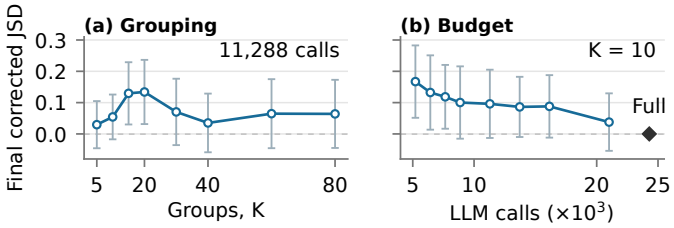


Figure 4. Separate five-seed sweeps. Circles show mean final corrected JSD, bars show reported 95% t-intervals, and the diamond denotes Full. Symmetric intervals may cross zero despite nonnegative JSD. Details: Appendices E.3.1 and E.3.2.

Table 6. Context probes: fraction of replies differing from recorded Full-LLM replies after APS values replace prompt fields. Mean: rounds $5, 10, \ldots, 50$ (500 preselected agents per round). Full details: Appendix E.2.2.

Replaced field	Mean	R30	R50
Previous state	0.155	0.210	0.080
Neighborhood	0.114	0.170	0.030
Both	0.198	0.270	0.060

shows recovery without sustained late-stage error growth, with metric summaries and trend estimates reported in Appendix E.2.1.

We test whether context differences change the next LLM decision. In recorded Full-LLM prompts, we replace the previous state, neighborhood summary, or both with APS values and keep all other inputs fixed. Table 6 reports response-flip rates,[6] falling from 0.210/0.170/0.270 at R30 to 0.080/0.030/0.060 at R50, respectively. The lower R50 rates indicate fewer decision disagreements after context substitution.

4.6 Grouping, Budget–Fidelity Trade-off, and Robustness (RQ5)

Figure 4a shows nonmonotone performance in a five-seed sweep over $K = 5$–80 at a fixed budget of 11,288 calls. Fewer, larger groups may contain agents with more diverse behaviors. More, smaller groups leave fewer prototype and audit calls per group under a fixed budget. The lowest observed corrected JSD and context TV occur at $K = 5$, with $K = 40$ close. The $K = 60/80$ settings have slightly higher exact match but larger distribution and context errors. Appendix E.3.1 reports the full fixed-budget comparison.

Figure 4b shows a separate five-seed sweep that fixes $K = 10$ and varies only the query budget. Comparing APS 5% with APS 70%, calls rise from 5,216 to 21,008. These percentages label budget settings, not the share of Full-LLM calls. Final corrected JSD falls from 0.1671 to 0.0378, and exact match rises from 0.6813 to 0.8062. The 40% setting uses fewer calls and achieves lower JSD and higher exact match than 50%. These separate sweeps motivate choosing both group number and query budget for the simulation setting and target fidelity metric. Appendix E.3.2 gives the full budget–fidelity analysis.

Table 7 reports compact 10K robustness checks across scenario prompts, LLM backends, and stratification backends, tested on matched agents against paired full-LLM references, and Figure 5 plots representative scenario trajectories. These checks support APS's approximation fidelity across the tested scenarios, LLM backends, and stratification methods. Appendix E.4.1 gives protocol notes. Additional economic-allocation, consumer-voucher, and mutual-aid tasks are reported in Appendix E.4.2.

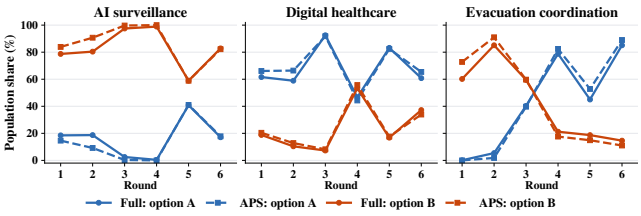


Figure 5. Multi-scenario trajectory visualization. Each panel tracks two scenario-specific focal options for full-LLM and APS rollouts; final distributional checks appear in Table 7.

5 Limitations

APS evaluates computational fidelity to an LLM-induced transition target, not behavioral validity as a model of human behavior. Biased or miscalibrated reference LLM outputs may therefore be preserved at scale: shadow-audit residual correction and tail-protected singleton routing reduce approximation

[6]The rate is the fraction of probe responses differing from the recorded Full-LLM decision. A rate of 0.080 means 8% differ. See Appendix E.2.2 for the protocol.

Table 7. Scenario, LLM backend, and stratification backend robustness. Scenario and stratification rows use glm-4.7-flash. LLM backend rows report means over the three 10K scenario checks. Grouping variants share the same 10K Full-LLM reference. Metrics are reference-aligned within each check. Brackets are 95% intervals.

Check	Variant	JSD ↓	JSD 95% CI	Exact match ↑	Exact 95% CI
Scenario	Polarized AI surveillance	0.004	[0.001, 0.016]	0.820	[0.812, 0.827]
Scenario	Minority-sensitive digital healthcare	0.027	[0.008, 0.067]	0.632	[0.623, 0.641]
Scenario	Coordination evacuation	0.025	[0.008, 0.053]	0.850	[0.843, 0.857]
LLM backend	deepseek/deepseek-v4-flash	0.007	[0.005, 0.011]	0.806	[0.801, 0.810]
LLM backend	google/gemini-3-flash-preview	0.008	[0.006, 0.011]	0.832	[0.828, 0.836]
LLM backend	openai/gpt-5.4-nano	0.004	[0.002, 0.007]	0.753	[0.748, 0.758]
LLM backend	qwen/qwen3.6-flash	0.002	[0.001, 0.004]	0.805	[0.800, 0.809]
LLM backend	x-ai/grok-4.1-fast	0.007	[0.005, 0.010]	0.849	[0.845, 0.853]
LLM backend	xiaomi/mimo-v2-omni	0.011	[0.008, 0.015]	0.656	[0.651, 0.661]
Stratification	Gaussian mixture, fixed K (14 strata)	0.055	[0.044, 0.070]	0.609	[0.599, 0.619]
Stratification	BIRCH, fixed K (14 strata)	0.050	[0.039, 0.064]	0.592	[0.582, 0.602]
Stratification	BIRCH, threshold only (32 strata)	0.039	[0.029, 0.050]	0.594	[0.584, 0.604]

bias relative to the online transition oracle, but do not validate that oracle. Within APS, static-profile grouping may smooth distinct responses when nearby agents have different latent histories. Audits and allocation respond to observed residuals, with their scope discussed in Appendix C.5. Although independent full-LLM references avoid leakage from APS states, external validity remains limited by WVS-derived seed records, perturbed profiles, generated social graphs, and fixed scenario prompts. The experiments simulate large populations under hypothetical urban scenarios, rather than calibrated populations of a specific city. The current population, graph, and prompts do not encode city-specific census composition, neighborhood geography, mobility or commuting flows, local media exposure, or institutional histories. Broader domains, longer horizons, richer and geographically grounded networks, and empirical human or city-level observational data remain important future tests; Appendix G discusses dual-use risks, subgroup reporting, and governance requirements.

6 Discussion and Conclusion

Discussion. APS frames population-scale LLM-agent simulation as recurrent LLM-oracle allocation. Under a fixed online-call budget, it keeps the specified LLM as the transition oracle and uses queried prototypes as same-round anchors for local response surfaces. Singleton-tail routing directly queries selected agents with poor prototype coverage, while shadow audits estimate propagation residuals for aggregate correction and later allocation. These mechanisms act on different parts of the simulation. The soft reporting ledger provides the audit-corrected aggregate $\widehat{p}_t^{\mathrm{audit}}$, while hard recurrent states determine later prompts. Because aggregate correction does not overwrite these states, fidelity checks should include hard-state exact match and neighborhood context alongside marginal distributions. Multi-round evaluation and context probes measure this distinction against independent full-LLM references. Grouping and budget should then be chosen together for the target metric and available resources, counting prototype, tail, and audit calls. The sweeps show why this choice should use measured results rather than assume that more groups or calls always improve fidelity. The dynamic-memory pilot in Appendix B.5 uses accumulated choice histories to refresh clusters, prototypes, and tails, motivating further study of richer memories and evolving agent states. Scalable simulators should therefore state their computational target and assumptions, report residual and tail-coverage diagnostics, and retain independent references and positive-probability audits where feasible.

Conclusion. We presented APS, a bias-controlled framework for population-scale LLM-agent simulation. APS keeps the LLM as the online transition oracle, queries adaptive core prototypes and singleton-tail agents, and uses shadow-audit residuals to correct population estimates and guide future budget allocation. Counting core-prototype, singleton-tail, and shadow-audit calls, APS reduces full-simulation cost from $O(NT)$ to $O(T(N^{1-\lambda} + M_{\mathrm{tot}}(N)))$. Empirically, APS scales to a 10M-agent, 8-round public-opinion simulation with 209.9K online LLM calls, a $381.1\times$ reduction over full simulation, and final-round JSD 0.094 against the corresponding full-LLM reference. Same-budget and strengthened-baseline comparisons support APS's approximation gains, while matched-budget ablations show the value of its components. End-to-end runtime remains lower than Full after all component overhead is included. The paired 50-round evaluation shows late-stage recovery in state and graph-context fidelity without sustained error growth. Robustness checks support fidelity across the tested scenarios, LLM backends, and stratification methods. We further explore grouping granularity, budget–fidelity trade-offs, cross-task robustness, and dynamic-memory extensions.

Acknowledgments and Disclosure of Funding

This work is supported by Zhongguancun Academy (Grant No. C20250521). The authors declare no competing interests.

References

Suhaib Abdurahman, Alireza Salkhordeh Ziabari, Alexander K Moore, Daniel M Bartels, and Morteza Dehghani. A primer for evaluating large language models in social-science research. *Advances in Methods and Practices in Psychological Science*, 8(2):25152459251325174, 2025.

Gati V Aher, Rosa I Arriaga, and Adam Tauman Kalai. Using large language models to simulate multiple humans and replicate human subject studies. In *International conference on machine learning*, pages 337–371. PMLR, 2023.

Anastasios N Angelopoulos, Stephen Bates, Clara Fannjiang, Michael I Jordan, and Tijana Zrnic. Prediction-powered inference. *Science*, 382(6671):669–674, 2023.

Anastasios N Angelopoulos, John C Duchi, and Tijana Zrnic. Ppi++: Efficient prediction-powered inference. *The Annals of Applied Statistics*, 20(3):2235–2252, 2026.

Lisa P Argyle, Ethan C Busby, Nancy Fulda, Joshua R Gubler, Christopher Rytting, and David Wingate. Out of one, many: Using language models to simulate human samples. *Political Analysis*, 31(3):337–351, 2023.

Ariel Flint Ashery, Luca Maria Aiello, and Andrea Baronchelli. Emergent social conventions and collective bias in llm populations. *Science Advances*, 11(20):eadu9368, 2025.

Christopher A Bail. Can generative ai improve social science? *Proceedings of the National Academy of Sciences*, 121(21):e2314021121, 2024.

Anat Bardi and Shalom H Schwartz. Values and behavior: Strength and structure of relations. *Personality and social psychology bulletin*, 29(10):1207–1220, 2003.

Alberto Bemporad. Active learning for regression by inverse distance weighting. *Information Sciences*, 626:275–292, 2023.

James Bisbee, Joshua D Clinton, Cassy Dorff, Brenton Kenkel, and Jennifer M Larson. Synthetic replacements for human survey data? the perils of large language models. *Political Analysis*, 32 (4):401–416, 2024.

Claudio Castellano, Santo Fortunato, and Vittorio Loreto. Statistical physics of social dynamics. *Reviews of modern physics*, 81(2):591–646, 2009.

Lingjiao Chen, Matei Zaharia, and James Zou. Frugalgpt: How to use large language models while reducing cost and improving performance. *arXiv preprint arXiv:2305.05176*, 2023a.

Weize Chen, Yusheng Su, Jingwei Zuo, Cheng Yang, Chenfei Yuan, Chi-Min Chan, Heyang Yu, Yaxi Lu, Yi-Hsin Hung, Chen Qian, et al. Agentverse: Facilitating multi-agent collaboration and exploring emergent behaviors. In *The Twelfth International Conference on Learning Representations*, 2023b.

Yiting Chen, Tracy Xiao Liu, You Shan, and Songfa Zhong. The emergence of economic rationality of gpt. *Proceedings of the National Academy of Sciences*, 120(51):e2316205120, 2023c.

Ayush Chopra. Large population models. *arXiv preprint arXiv:2507.09901*, 2025.

William Gemmell Cochran. *Sampling techniques*. 1977.

Jyotishka Datta and Nicholas G Polson. Prediction-powered inference with inverse probability weighting. *arXiv preprint arXiv:2508.10149*, 2025.

Thomas Davidson and Daniel Karell. Integrating generative artificial intelligence into social science research: Measurement, prompting, and simulation. *Sociological Methods & Research*, 54(3): 775–793, 2025.

Shizhe Diao, Pengcheng Wang, Yong Lin, Rui Pan, Xiang Liu, and Tong Zhang. Active prompting with chain-of-thought for large language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1330–1350, 2024.

Abdourahmane Diaw, Michael McKerns, Irina Sagert, Liam G Stanton, and Michael S Murillo. Efficient learning of accurate surrogates for simulations of complex systems. *Nature Machine Intelligence*, 6(5):568–577, 2024.

Danica Dillion, Niket Tandon, Yuling Gu, and Kurt Gray. Can ai language models replace human participants? *Trends in Cognitive Sciences*, 27(7):597–600, 2023.

Maik Döring, László Györfi, and Harro Walk. Rate of convergence of k-nearest-neighbor classification rule. *Journal of Machine Learning Research*, 18(227):1–16, 2018.

Joshua M Epstein and Robert Axtell. *Growing artificial societies: social science from the bottom up*. 1996.

Dan Feldman. Core-sets: An updated survey. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 10(1):e1335, 2020.

E Frantar, S Ashkboos, T Hoefler, and D Alistarh. Optq: Accurate quantization for generative pre-trained transformers. 2023. 2023.

Chen Gao, Xiaochong Lan, Zhihong Lu, Jinzhu Mao, Jinghua Piao, Huandong Wang, Depeng Jin, and Yong Li. S3: Social-network simulation system with large language model-empowered agents. *arXiv preprint arXiv:2307.14984*, 2023.

Chen Gao, Xiaochong Lan, Nian Li, Yuan Yuan, Jingtao Ding, Zhilun Zhou, Fengli Xu, and Yong Li. Large language models empowered agent-based modeling and simulation: A survey and perspectives. *Humanities and Social Sciences Communications*, 11(1):1–24, 2024.

Igor Grossmann, Matthew Feinberg, Dawn C Parker, Nicholas A Christakis, Philip E Tetlock, and William A Cunningham. Ai and the transformation of social science research. *Science*, 380(6650): 1108–1109, 2023.

Haoxiang Guan, Jiyan He, Liyang Fan, Zhenzhen Ren, Shaobin He, Xin Yu, Yuan Chen, Shuxin Zheng, Tie-Yan Liu, and Zhen Liu. Modeling earth-scale human-like societies with one billion agents. *arXiv preprint arXiv:2506.12078*, 2025.

Taicheng Guo, Xiuying Chen, Yaqi Wang, Ruidi Chang, Shichao Pei, Nitesh V Chawla, Olaf Wiest, and Xiangliang Zhang. Large language model based multi-agents: A survey of progress and challenges. 2024.

Christian Haerpfer, Ronald Inglehart, Alejandro Moreno, Christian Welzel, Kseniya Kizilova, Jaime Diez-Medrano, Marta Lagos, Pippa Norris, Eduard Ponarin, and Bi Puranen. World values survey wave 7 (2017-2022) cross-national data-set, 2022.

James J Heckman. Heterogeneity and state dependence. In *Studies in labor markets*, pages 91–140. University of Chicago Press, 1981.

Sirui Hong, Mingchen Zhuge, Jonathan Chen, Xiawu Zheng, Yuheng Cheng, Jinlin Wang, Ceyao Zhang, Zili Wang, Steven Ka Shing Yau, Zijuan Lin, et al. Metagpt: Meta programming for a multi-agent collaborative framework. In *The twelfth international conference on learning representations*, 2023.

John J Horton, Apostolos Filippas, and Benjamin S Manning. Large language models as simulated economic agents: What can we learn from homo silicus? 2023.

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

Na Jiang, Fuzhen Yin, Boyu Wang, and Andrew T Crooks. A large-scale geographically explicit synthetic population with social networks for the united states. *Scientific Data*, 11(1):1204, 2024.

Alan P Kirman. Whom or what does the representative individual represent? *Journal of economic perspectives*, 6(2):117–136, 1992.

Dan M Kluger and Stephen Bates. M-estimation under two-phase multiwave sampling with applications to prediction-powered inference. *arXiv preprint arXiv:2602.16933*, 2026.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the 29th symposium on operating systems principles*, pages 611–626, 2023.

Yaniv Leviathan, Matan Kalman, and Yossi Matias. Fast inference from transformers via speculative decoding. In *International Conference on Machine Learning*, pages 19274–19286. PMLR, 2023.

Guohao Li, Hasan Hammoud, Hani Itani, Dmitrii Khizbullin, and Bernard Ghanem. Camel: Communicative agents for" mind" exploration of large language model society. volume 36, pages 51991–52008, 2023.

Yexiong Lin, Shanshan Ye, Yu Yao, Zhen Fang, Bo Han, and Tongliang Liu. Squidagent: Parallelize wisely, coordinate efficiently. *arXiv preprint arXiv:2610.08647*, 2026.

Zhicheng Lin. Six fallacies in substituting large language models for human participants. *Advances in Methods and Practices in Psychological Science*, 8(3):25152459251357566, 2025.

Stuart Lloyd. Least squares quantization in pcm. *IEEE transactions on information theory*, 28(2): 129–137, 1982.

Michael W Macy and Robert Willer. From factors to actors: Computational sociology and agent-based modeling. *Annual review of sociology*, 28(1):143–166, 2002.

Benjamin S Manning, Kehang Zhu, and John J Horton. Automated social science: Language models as scientist and subjects. 2024.

Daniel McFadden. The choice theory approach to market research. *Marketing science*, 5(4):275–297, 1986.

Xinyi Mou, Xuanwen Ding, Qi He, Liang Wang, Jingcong Liang, Xinnong Zhang, Libo Sun, Jiayu Lin, Jie Zhou, Huang Xuanjing, et al. From individual to society: A survey on social simulation driven by large language model-based agents. *ACM Computing Surveys*, 2024.

Reagan Mozer, Nicole E Pashley, and Luke Miratrix. Stratified sampling for model-assisted estimation with surrogate outcomes. *arXiv preprint arXiv:2602.12992*, 2026.

SP Mukherjee. Stratified sampling: some associated problems. *Calcutta Statistical Association Bulletin*, 75(1):48–59, 2023.

Joon Sung Park, Joseph O'Brien, Carrie Jun 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*, pages 1–22, 2023.

Joon Sung Park, Carolyn Q Zou, Aaron Shaw, Benjamin Mako Hill, Carrie Cai, Meredith Ringel Morris, Robb Willer, Percy Liang, and Michael S Bernstein. Generative agent simulations of 1,000 people. *arXiv preprint arXiv:2411.10109*, 52, 2024.

Jinghua Piao, Yuwei Yan, Jun Zhang, Nian Li, Junbo Yan, Xiaochong Lan, Zhihong Lu, Zhiheng Zheng, Jing Yi Wang, Di Zhou, et al. Agentsociety: Large-scale simulation of llm-driven generative agents advances understanding of human behaviors and society. 2025.

Manon Prédhumeau and Ed Manley. A synthetic population for agent-based modelling in canada. *Scientific Data*, 10(1):148, 2023.

Yao Qu and Jue Wang. Performance and biases of large language models in public opinion simulation. *Humanities and Social Sciences Communications*, 11(1):1–13, 2024.

Carl-Erik Särndal, Bengt Swensson, and Jan Wretman. *Model assisted survey sampling*. 2003.

Shalom H Schwartz. Universals in the content and structure of values: Theoretical advances and empirical tests in 20 countries. In *Advances in experimental social psychology*, volume 25, pages 1–65. Elsevier, 1992.

Burr Settles. *Active learning literature survey*. 2009.

Ying Sheng, Lianmin Zheng, Binhang Yuan, Zhuohan Li, Max Ryabinin, Beidi Chen, Percy Liang, Christopher Ré, Ion Stoica, and Ce Zhang. Flexgen: High-throughput generative inference of large language models with a single gpu. In *International Conference on Machine Learning*, pages 31094–31116. PMLR, 2023.

Charles J Stone. Consistent nonparametric regression. *The annals of statistics*, pages 595–620, 1977.

Jiakai Tang, Heyang Gao, Xuchen Pan, Lei Wang, Haoran Tan, Dawei Gao, Yushuo Chen, Xu Chen, Yankai Lin, Yaliang Li, et al. Gensim: A general social simulation platform with large language model based agents. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (System Demonstrations)*, pages 143–150, 2025.

Robert P Tett and Hal A Guterman. Situation trait relevance, trait expression, and cross-situational consistency: Testing a principle of trait activation. *Journal of research in Personality*, 34(4): 397–423, 2000.

Alexander Sasha Vezhnevets, John P Agapiou, Avia Aharon, Ron Ziv, Jayd Matyas, Edgar A Duéñez-Guzmán, William A Cunningham, Simon Osindero, Danny Karmon, and Joel Z Leibo. Generative agent-based modeling with actions grounded in physical, social, or digital space using concordia. *arXiv preprint arXiv:2312.03664*, 2023.

Angelina Wang, Jamie Morgenstern, and John P Dickerson. Large language models that replace human participants can harmfully misportray and flatten identity groups. *Nature Machine Intelligence*, 7(3):400–411, 2025.

Lei Wang, Chen Ma, Xueyang Feng, Zeyu Zhang, Hao Yang, Jingsen Zhang, Zhiyuan Chen, Jiakai Tang, Xu Chen, Yankai Lin, et al. A survey on large language model based autonomous agents. *Frontiers of Computer Science*, 18(6):186345, 2024.

Robert Wieczorkowski, Wojciech WĂłjciak, et al. Recursive neyman algorithm for optimum sample allocation under box constraints on sample sizes in strata. *arXiv preprint arXiv:2304.07034*, 2023.

Changbao Wu and Mary E. Thompson. *Sampling Theory and Practice*. ICSA Book Series in Statistics. Springer, Cham, 2020.

Zhiheng Xi, Wenxiang Chen, Xin Guo, Wei He, Yiwen Ding, Boyang Hong, Ming Zhang, Junzhe Wang, Senjie Jin, Enyu Zhou, et al. The rise and potential of large language model based agents: A survey. *Science China Information Sciences*, 68(2):121101, 2025.

Mengzhou Xia, Sadhika Malladi, Suchin Gururangan, Sanjeev Arora, and Danqi Chen. Less: Selecting influential data for targeted instruction tuning. 2024.

Yue Xing, Qifan Song, and Guang Cheng. Benefit of interpolation in nearest neighbor algorithms. *SIAM Journal on Mathematics of Data Science*, 4(2):935–956, 2022.

Yuwei Xu, Shulun Zhang, Yingli Zhou, Shipei Zeng, Laks VS Lakshmanan, and Chenhao Ma. Topology-aware llm-driven social simulation: A unified framework for efficient and realistic agent dynamics. *arXiv preprint arXiv:2604.18011*, 2026.

Yuwei Yan, Qingbin Zeng, Zhiheng Zheng, Jingzhe Yuan, Jie Feng, Jun Zhang, Fengli Xu, and Yong Li. Opencity: A scalable platform to simulate urban activities with massive llm agents. *arXiv preprint arXiv:2410.21286*, 2024.

Ziyi Yang, Zaibin Zhang, Zirui Zheng, Yuxian Jiang, Ziyue Gan, Zhiyu Wang, Zijian Ling, Jinsong Chen, Martz Ma, Bowen Dong, et al. Oasis: Open agent social interaction simulations with one million agents. *arXiv preprint arXiv:2411.11581*, 2024.

Chunxu Zhang, Ximing Li, Hongbin Pei, Zijian Zhang, Bing Liu, and Bo Yang. Laennet: Learning robust gcns by propagating labels. *Neural Networks*, 168:652–664, 2023.

Xuhui Zhou, Hao Zhu, Leena Mathur, Ruohong Zhang, Haofei Yu, Zhengyang Qi, Louis-Philippe Morency, Yonatan Bisk, Daniel Fried, Graham Neubig, et al. Sotopia: Interactive evaluation for social intelligence in language agents. 2023.

Xiaojin Zhu, Zoubin Ghahramani, and John D Lafferty. Semi-supervised learning using gaussian fields and harmonic functions. In *Proceedings of the 20th International conference on Machine learning (ICML-03)*, pages 912–919, 2003.

Caleb Ziems, William Held, Omar Shaikh, Jiaao Chen, Zhehao Zhang, and Diyi Yang. Can large language models transform computational social science? *Computational Linguistics*, 50(1): 237–291, 2024.

Tijana Zrnic and Emmanuel J Candès. Active statistical inference. *arXiv preprint arXiv:2403.03208*, 2024.

Appendix

Contents

A Additional Related Work . . . **17**

B APS Algorithms, Diagnostics, and Dynamic Extensions . . . **18**

B.1 Detailed APS Algorithms . . . 18

B.2 Shadow-Audit Diagnostic Definitions . . . 18

B.3 Scale Schedules . . . 19

B.4 Scaling-Table APS Settings . . . 20

B.5 Dynamic-Memory Agents and Refreshed Tails . . . 20

C Theory Details and Error Decomposition . . . **21**

C.1 Query Complexity . . . 21

C.2 Design Validity of Audit Correction . . . 21

C.3 Connection to the Full-Reference Target . . . 21

C.4 Residual Variance Reduction . . . 22

C.5 Feature–Behavior Assumption: Basis and Scope . . . 22

C.6 Local Response-Surface Regularity and Base Error . . . 22

C.7 Tail-Protected Singleton Routing . . . 23

C.8 Multi-Round Decomposition . . . 23

C.9 Residual-Aware Allocation . . . 23

C.10 Metric Connection . . . 23

D Experimental Protocol and Reproducibility . . . **23**

D.1 Population and Simulation Inputs . . . 24

D.2 Baseline Implementations . . . 25

D.3 Evaluation Protocols . . . 27

D.4 Resource Accounting . . . 28

E Extended Evaluation . . . **29**

E.1 Strengthened Baselines under Matched Conditions . . . 29

E.2 Multi-Round Stability . . . 30

E.3 Grouping and Budget Sensitivity . . . 31

E.4 Robustness and Cross-Task Evaluation . . . 32

F Prompt Templates and Scenario Text . . . **33**

F.1 Generic LLM Decision Template . . . 33

F.2 Main and Ablation Scenario: Subway Chemical Attack . . . 34

F.3 Scenario-Generalization Prompts . . . 35

G Ethics and Social Impact . . . **36**

Appendix roadmap. Appendix A provides related-work context. Appendix B presents the algorithms, diagnostics, scale settings, and dynamic-memory extension, followed by assumptions and derivations in Appendix C. Appendix D organizes the experimental protocols by inputs, baselines, evaluation, and resources. Appendix E reports strengthened baseline comparisons, multi-round stability, grouping and budget sensitivity, and cross-task robustness. Exact prompts and use boundaries follow in Appendices F and G.

A Additional Related Work

Prediction-powered inference, adaptive acquisition, and grouped-agent simulation provide related foundations [Angelopoulos et al., 2023, 2026, Zrnic and Candès, 2024, Yan et al., 2024]. APS formulates recurrent approximation as an audited LLM-oracle allocation problem and contributes a bias-controlled adaptive prototype framework for recurrent social simulation under a limited LLM budget.

LLM agents as a simulation substrate. Recent surveys and perspective articles describe a broader shift from using LLMs as task solvers to using them as components in synthetic social worlds, behavioral experiments, and computational social-science workflows [Xi et al., 2025, Wang et al., 2024, Guo et al., 2024, Grossmann et al., 2023, Bail, 2024, Davidson and Karell, 2025, Ziems et al., 2024, Gao et al., 2024, Mou et al., 2024]. Early generative-agent and multi-agent systems focus on believable memory, role play, interaction, social intelligence, and group collaboration [Park et al., 2023, Li et al., 2023, Chen et al., 2023b, Hong et al., 2023, Zhou et al., 2023, Vezhnevets et al., 2023]. This line establishes that LLM agents can produce coherent local behavior, but it also shifts the methodological question from whether agents can act plausibly in small scenes to whether population-level trajectories remain valid, calibrated, and affordable when many agents interact over time.

Interdisciplinary behavioral and social simulation. LLM-based simulation now intersects psychology, political science, economics, sociology, and computational social science. Studies use LLM personas or agent populations to replicate human-subject experiments, survey-like responses, public opinion, economic choice, trust, and convention formation [Argyle et al., 2023, Aher et al., 2023, Dillion et al., 2023, Horton et al., 2023, Chen et al., 2023c, Qu and Wang, 2024, Park et al., 2024, Ashery et al., 2025]. At the same time, social-science and psychology work warns that synthetic respondents should not be treated as human truth: LLM outputs can be biased, overconfident, prompt-sensitive, temporally stale, or weak at preserving subgroup heterogeneity and causal structure [Bisbee et al., 2024, Abdurahman et al., 2025, Wang et al., 2025, Lin, 2025]. APS follows the more conservative view that LLM simulations are model-based instruments requiring explicit reference definitions, validation metrics, and error controls rather than replacements for human evidence.

Population-scale platforms and decision-support use cases. The practical motivation for large LLM-agent simulation is increasingly framed around scenario screening, public-opinion analysis, platform governance, policy prototyping, crisis response, and other "what-if" uses where researchers or decision makers need macro-level trajectories before acting in the real world [Manning et al., 2024, Qu and Wang, 2024, Piao et al., 2025]. Scale-oriented systems simulate social networks, online platforms, synthetic populations, and large societies using general simulation engines, social-media environments, learned or optimized surrogates, differentiable population frameworks, or topology-aware grouping [Gao et al., 2023, Tang et al., 2025, Yang et al., 2024, Prédhumeau and Manley, 2023, Jiang et al., 2024, Guan et al., 2025, Chopra, 2025, Xu et al., 2026]. These systems broaden the application space, but they also make cost–fidelity accounting central: a result meant to inform real decisions must report what was queried, what was propagated, what reference it approximates, and where approximation bias may enter.

Scaling strategies and their statistical risks. Scaling LLM-agent simulation can reduce cost through model-serving optimizations, smaller or learned surrogates, grouped representatives, static or topology-aware propagation, and platform-level batching [Frantar et al., 2023, Leviathan et al., 2023, Chen et al., 2023a, Sheng et al., 2023, Kwon et al., 2023, Guan et al., 2025, Xu et al., 2026]. OpenCity scales grouped agents through attribute grouping, group-and-distill prompting, and scheduling [Yan et al., 2024]. At the execution level, SquidAgent improves multi-agent efficiency through selective parallelization and reduced coordination overhead [Lin et al., 2026]. These strategies address different parts of the cost problem, but they can also introduce distinct statistical risks: removing the online

LLM anchor changes the transition rule, grouping agents can smooth isolated or heterogeneous regions, and recurrent propagation can accumulate context mismatch across rounds. APS is closest to the scale-oriented simulation line, but differs by preserving the specified LLM as the online transition oracle and pairing prototype propagation with online residual correction, tail-protected singleton routing, and explicit call accounting.

Prototype propagation and bias control. Representative-agent models, coresets, and graph-based label propagation show how sparse labels or representative points can summarize larger populations [Kirman, 1992, Feldman, 2020, Zhu et al., 2003, Zhang et al., 2023]. These approaches are effective when local structure is informative, but their usual objective is static approximation or label completion rather than recurrent LLM-agent simulation. APS draws on this prototype view in a graph-conditioned, recurrent setting where propagated hard states condition later prompts. Its integrated audited framework pairs local propagation with tail-protected singleton routing, shadow-audit residual correction, and a soft reporting ledger distinct from the hard recurrent ledger.

Auditing, model-assisted estimation, and adaptive allocation. Model-assisted survey estimation uses a prediction model to form an initial population estimate and probability samples to estimate the residual [Horvitz and Thompson, 1952, Cochran, 1977, Särndal et al., 2003]. Prediction-powered inference and PPI++ similarly combine population-wide predictions with residuals from a labeled sample [Angelopoulos et al., 2023, 2026]. Inverse-probability weighting accommodates unequal sampling probabilities [Datta and Polson, 2025]. Conditional on the APS history at a fixed round, shadow-audit correction follows this predict-then-debias structure. APS rebuilds its proxy from current-round LLM responses. Shadow labels correct the reported aggregate and guide future allocation without overwriting hard recurrent states.

Classical stratified and Neyman allocation study how to distribute samples across strata, with recent work extending allocation under practical constraints [Mukherjee, 2023, Wieczorkowski et al., 2023]. Active statistical inference, residual-variance allocation, and multiwave sampling use prediction quality or earlier observations to allocate expensive labels [Zrnic and Candès, 2024, Mozer et al., 2026, Kluger and Bates, 2026]. LLM work on active prompting and influential data selection similarly motivates selective querying under limited oracle budgets [Settles, 2009, Diao et al., 2024, Xia et al., 2024]. APS integrates allocation with prototype and tail queries that advance the simulation, while audits evaluate propagation and guide later calls.

B APS Algorithms, Diagnostics, and Dynamic Extensions

This appendix presents detailed pseudocode, shadow-audit diagnostics, scale schedules, and the APS settings used in Table 1. It then examines how the grouping, prototypes, and tails can be refreshed when agents' recorded histories change. The corresponding assumptions and derivations are given in Appendix C.

B.1 Detailed APS Algorithms

Algorithm 1 expands the main rollout loop. Algorithm 2 expands the shadow-audit residual correction used after prototype propagation. We keep them separate because the first algorithm defines the recurrent simulation path, whereas the second defines the estimator correction and next-round allocation diagnostics.

B.2 Shadow-Audit Diagnostic Definitions

For stratum C_m at round t, let $U_{m,t} \subset C_m \setminus S_t$ be a shadow-audit set sampled across predicted state groups, with extra coverage for rare states, and let $a_{m,t} = |U_{m,t}|$. The mismatch rate $\hat{e}_{m,t}$ is the fraction of audited agents whose APS hard state differs from the shadow-audit decision. For each category y, the residual is $\mathbb{I}[\tilde{y}_i^t = y] - h_i^t(y)$, where $h_i^t(y)$ is the local-response-surface probability. The residual-variance summary $\widehat{V}_{m,t}^{\mathrm{res}}$ sums the empirical variances of these residuals across categories. Monitoring JSD compares the APS-predicted and shadow-audit state distributions on the same audited agents.

The residual-risk score in the main text additionally uses $\hat{\rho}_{m,t}$, the average distance from shadow-audited non-prototypes to their queried-prototype support; $\widehat{L}_{m,t}$, a local disagreement-slope or

Algorithm 1 Detailed APS Main Simulation Loop

Input: agent features $\{x_i\}_{i=1}^N$, graph G, initial states y^0, scenario events $\{s_t\}_{t=1}^T$, option set $\mathcal{Y}$, LLM pipeline and parser, schedules $M_{\mathrm{core}}(N)$, $M_{\mathrm{out}}(N)$, $\alpha(N)$, shadow-audit schedule $A_t(N)$, support size κ, interpolation rule $\mathcal{I}$, random seed ξ.
Output: hard recurrent states $\{\hat{y}^t\}_{t=1}^T$, projected shadow-audit-corrected distributions, diagnostics, rollout-query records, and shadow-audit records.
Set seed ξ; initialize $\hat{y}_i^0 = y_i^0$ and stratum scores $R_{m,0} = 1$.
Compute tail scores s_i; select singleton-tail set O_N of size $M_{\mathrm{out}}(N)$.
Partition $\mathcal{A}_N \setminus O_N$ into $M_{\mathrm{core}}(N)$ core strata $\{C_m\}$.
for $t = 1$ to T **do**
  Build neighbor summaries $\hat{g}_i^t = \Gamma_i(G, \hat{y}^{t-1})$ for all agents.
  Compute the round core-prototype budget from $\alpha(N)N$ and allocate stratum budgets $\{B_{m,t}\}$ using $|C_m|$, previous scores $R_{m,t-1}$, and the minimum-one rounding rule.
  Select prototypes $S_{m,t} \subset C_m$ with $|S_{m,t}| = B_{m,t}$; set $S_t = \cup_m S_{m,t}$ and direct rollout set $D_t = S_t \cup O_N$.
  Query the LLM for every $i \in D_t$ using profile x_i, previous state $\hat{y}_i^{t-1}$, neighbor summary $\hat{g}_i^t$, and event s_t; parse the response into an option in $\mathcal{Y}$.
  Set h_i^t to the one-hot parsed decision and $\hat{y}_i^t$ to that decision for every directly queried agent $i \in D_t$.
  for each core stratum C_m **do**
    for each non-prototype agent $j \in C_m \setminus S_{m,t}$ **do**
      Find the κ nearest queried prototypes in $S_{m,t}$ under the implementation distance.
      Form h_j^t by interpolation rule $\mathcal{I}$ over their one-hot decisions.
      Set $\hat{y}_j^t = \arg\max_{y \in \mathcal{Y}} h_j^t(y)$ with deterministic tie-breaking.
    end for
  end for
  Run Algorithm 2 on the non-prototype core correction frame.
  Store hard states, projected audit-corrected distribution, diagnostics, and call records.
end for

Algorithm 2 Detailed Shadow-Audit Correction and Score Update

Input: soft vectors h^t, previous hard states $\hat{y}^{t-1}$, graph summaries $\hat{g}^t$, correction frame $F_t = \cup_m (C_m \setminus S_{m,t})$, shadow-audit budget $A_t(N)$, shadow-audit rule, and diagnostic weights.
Output: unprojected and projected shadow-audit-corrected distributions, shadow-audit diagnostics, inclusion probabilities, and next-round scores $R_{m,t}$.
Allocate shadow-audit counts $a_{m,t}$ across core strata, respecting $0 \le a_{m,t} \le |F_t \cap C_m|$ and $\sum_m a_{m,t} \le A_t(N)$.
for each core stratum C_m **do**
  Select audit agents $U_{m,t} \subseteq F_t \cap C_m$ using the recorded audit design, with inclusion probabilities $\psi_{i,t} > 0$.
  Query each $i \in U_{m,t}$ with the same LLM, parser, event s_t, and APS context $(x_i, \hat{y}_i^{t-1}, \hat{g}_i^t)$ used by rollout calls.
  Parse the shadow decision $\tilde{y}_i^t$ and record retry or repair metadata.
  Compute residual vectors $\mathbb{I}[\tilde{y}_i^t = y] - h_i^t(y)$ for all $y \in \mathcal{Y}$.
  Summarize mismatch, residual variance, monitoring JSD, prototype-support distance, local disagreement or curvature, and rare-state recall.
  Update $R_{m,t}$ from the normalized diagnostics using Eq. 6.
end for
Form the unprojected audit-corrected estimator by Eq. 5.
Clip negative entries and renormalize to the simplex for JSD reporting; keep the unprojected vector for the design-validity analysis.
Return both estimates, diagnostics, inclusion probabilities, and updated scores.

curvature estimate; and $\hat{r}_{m,t}$, the fraction of shadow-audit rare-state labels in $U_{m,t}$ that are also recovered by APS.

B.3 Scale Schedules

The default core-stratum schedule is

$$M_{\mathrm{core}}(N) = \begin{cases} M_b, & N \le N_b, \\ \left\lfloor M_b \left(\frac{N}{N_b}\right)^{\eta} \right\rfloor, & N > N_b, \end{cases} \qquad N_b = 5000, \quad M_b = 10, \quad 0 < \eta < 1. \tag{9}$$

The concrete tail schedule used in our experiments is

$$M_{\text{out}}(N) = \begin{cases} \lceil \delta_0 N_b \rceil, & N \le N_b, \\ \left\lceil \delta_0 N_b \left(\frac{N}{N_b} \right)^{\zeta} \right\rceil, & N > N_b, \end{cases} \qquad \delta_0 = 0.05, \quad 0 < \zeta < 1. \tag{10}$$

The number of singleton-tail queries grows as $O(N^{\zeta})$, while the routed fraction shrinks as $O(N^{\zeta-1})$. This fraction is an algorithmic coverage budget, not an estimate of feature-space tail prevalence.

The shadow-audit budget $A_t(N)$ is $\gamma N_b \max\{1, (N/N_b)^{\beta_a}\}$ rounded down, with a minimum of $A_{\min}$ calls. Here $0 < \beta_a < 1$ controls call growth and γ sets its coefficient. Experiments set $\gamma = 0.05$, $\beta_a = 1 - \lambda$, and $A_{\min} = 1$.

For the residual-aware allocation in Eq. 4, the minimum-one rule assigns at least one core prototype to each nonempty core stratum whenever the nominal core-prototype budget permits; the remaining slots are distributed according to the residual-aware stratum weights and then rounded to integers.

B.4 Scaling-Table APS Settings

Table 8 gives the concrete APS settings behind the three superscripts in Table 1. All three use $T = 8$, seed 42, robust-MAD tail scoring, mini-batch or streaming-compatible K-means for core strata, inverse-distance interpolation with $\kappa = 5$, initial scores $R_{m,0} = 1$, and the residual-aware allocation rule in Eq. 6 with normalized diagnostics, $\lambda_\rho = \lambda_e = \lambda_r = 1$, and $\tau = 10^{-6}$.

Table 8. Concrete APS schedule settings for Table 1. Core rate determines the pre-round core-prototype budget before residual-aware allocation across core strata. Reported calls are completed online calls across eight rounds and include core-prototype, singleton-tail, and shadow-audit queries.

Sup.	Table rows	Core rate	M_{core}	M_{out}	Shadow-audit A_t	Calls
†	3K	fixed 0.20	10	250	250	8.9K
‡	10K, 100K	fixed 0.20	$\lfloor 10(N/5000)^{0.5} \rfloor$	$\lceil 250(N/5000)^{0.4} \rceil$	$\max\{1, \lfloor 250(N/5000)^{0.4} \rfloor\}$	20.8K; 172.0K
⋆	1M, 10M	$0.15(5000/N)^{0.6}$	$\lfloor 10(N/5000)^{0.5} \rfloor$	$\lceil 250(N/5000)^{0.4} \rceil$	$\max\{1, \lfloor 250(N/5000)^{0.4} \rfloor\}$	83.4K; 209.9K

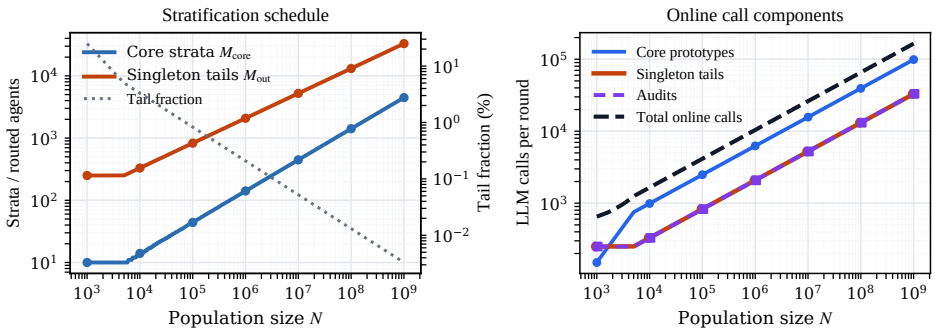


Figure 6. Scale-dependent APS schedules. Left: core strata grow as N^{η}, singleton-tail agents grow as N^{ζ}, and the routed fraction shrinks as $N^{\zeta-1}$. Right: per-round online LLM calls under the production schedule, counting core-prototype, singleton-tail, and shadow-audit queries.

B.5 Dynamic-Memory Agents and Refreshed Tails

We conduct a paired 1,000-agent, eight-round proof of concept. All queried prompts contain the complete prior choice history. Static-memory APS retains its fixed 19-dimensional profile representation, grouping, and singleton-tail set. Dynamic-memory APS appends each prior choice as a normalized ordinal feature, expanding the grouping representation from 19 to 26 dimensions by round 8. Before each round, it re-clusters agents, re-selects prototypes, and refreshes singleton-tail membership. Tail selection uses a fixed equal-weight combination of static-profile and accumulated-history outlier ranks.

Table 9. Dynamic-memory pilot with nearly matched APS call budgets. Full-LLM uses 8,000 calls.

Method	Calls	Mean JSD ($\times 10^{-3}$)	Final JSD ($\times 10^{-3}$)	Mean exact	Final exact
Static-memory APS	5,564	2.24	1.57	0.8445	0.905
Dynamic-memory APS	5,572	1.72	0.28	0.8470	0.914

Table 9 shows that dynamic-memory APS improves both aggregate and individual-level fidelity in this setting. On average, 6.9% of tail membership is newly selected after round 1, showing that direct-query coverage responds to divergent histories. This pilot establishes the feasibility of refreshing APS support as agents change. Extensions can use compressed free-text memories, interaction histories, changing goals, or learned latent states, while measuring the added embedding, clustering, and LLM-call costs.

C Theory Details and Error Decomposition

This appendix gives the derivations summarized in Section 3.

C.1 Query Complexity

For $N > N_b$, Eq. 3 gives a raw prototype budget of order $N^{1-\lambda}$. The minimum-one rule adds at most one query per nonempty core stratum. Each round also queries $M_{\mathrm{out}}(N)$ singleton-tail agents and $A(N)$ shadow-audit agents. Summing these costs over T rounds gives the query-complexity bound in Section 3. Core strata, tails, and audits grow with exponents η, ζ, and β_a, respectively, so the largest of these exponents and $1-\lambda$ determines the overall growth rate. When all are below one, calls grow sublinearly in N for fixed T. With $\beta_a = 1-\lambda$, audit and prototype calls have the same order.

C.2 Design Validity of Audit Correction

Fix round t and condition on the APS history $\mathcal{F}_{t-1}^{\mathrm{APS}}$. Select the direct-query set before observing current labels. Direct and shadow queries use the designated LLM at the APS-maintained context, with expected label vector $\boldsymbol{q}_{i,t}^{A}$. Shadow labels retain this expectation conditional on the direct-query set and constructed response surface. Given this information, audit selection has known positive inclusion probabilities $\psi_{i,t}$ and is independent of the not-yet-observed shadow labels. Directly queried agents contribute $\boldsymbol{q}_{i,t}^{A}$ in expectation. For a correction-frame agent, inverse-probability weighting gives the conditional identity

$$\mathbb{E}\left[\frac{\mathbb{I}[i \in U_t]}{\psi_{i,t}}(\tilde{\boldsymbol{Z}}_i^t - \boldsymbol{h}_i^t)\right] = \boldsymbol{q}_{i,t}^{A} - \boldsymbol{h}_i^t.$$

Adding the baseline $\boldsymbol{h}_i^t$ yields $\boldsymbol{q}_{i,t}^{A}$. Summing over direct and correction-frame units gives

$$\mathbb{E}[\widehat{\boldsymbol{p}}_t^{\mathrm{audit}} \mid \mathcal{F}_{t-1}^{\mathrm{APS}}] = \bar{\boldsymbol{p}}_t^{A}.$$

The response surface affects variance but need not be correct for unbiasedness. This result applies before clipping and renormalization, which can introduce finite-sample projection bias. Positive inclusion can be maintained by mixing adaptive selection with an exploration design that gives every correction-frame agent a nonzero chance of selection.

C.3 Connection to the Full-Reference Target

Audit correction targets transitions at APS-maintained contexts. The difference between the APS-context and full-reference targets is $\epsilon_{\mathrm{ctx},t} = \|\bar{\boldsymbol{p}}_t^{A} - \bar{\boldsymbol{p}}_t^{\star}\|_1$. Suppose the transition kernel is locally Lipschitz: changing context by distance d_z changes its response distribution by at most $L_t d_z$ in L_1. Averaging this bound over agents gives

$$\epsilon_{\mathrm{ctx},t} \le \frac{L_t}{N}\sum_{i=1}^{N} d_z(\hat{z}_i^t, z_i^{\star,t}), \qquad z_i^{\star,t} = (x_i, y_i^{\star,t-1}, g_i^{\star,t}).$$

Here $\hat{z}_i^t$ and $z_i^{\star,t}$ contain the profile, previous state, and neighborhood summary used by APS and Full-LLM. This gap requires separate assessment of recurrent states and graph context, beyond aggregate correction.

C.4 Residual Variance Reduction

Fix category y and write $Z_i(y) = \mathbb{I}[\tilde{y}_i^t = y]$, $h_i(y) = h_i^t(y)$, and $q_i(y) = q_{i,t}^A(y)$. Under independent Poisson shadow-audit sampling, conditional on realized shadow-audit labels and the response surface,

$$\mathrm{Var}_{U_t}(\widehat{p}_t^{\,\mathrm{audit}}(y) \mid \tilde{y}, h) = \frac{1}{N^2} \sum_{i \in F_t} \frac{1 - \psi_{i,t}}{\psi_{i,t}} (Z_i(y) - h_i(y))^2.$$

Setting $h_i(y) = 0$ gives the Horvitz–Thompson design variance. APS therefore replaces raw-label variation by residual variation, reducing this variance when the weighted squared residuals are smaller. If the shadow label is also modeled as a stochastic oracle draw, then for fixed $h_i(y)$,

$$\mathrm{Var}\left[h_i(y) + \frac{\mathbb{I}[i \in U_t]}{\psi_{i,t}}(Z_i(y) - h_i(y))\right] = \frac{q_i(y)(1 - q_i(y))}{\psi_{i,t}} + \left(\frac{1}{\psi_{i,t}} - 1\right)(q_i(y) - h_i(y))^2.$$

The first term is oracle variation under shadow-audit sampling; with deterministic decoding it is zero for point-mass transitions. The second term is residual response-surface error.

C.5 Feature–Behavior Assumption: Basis and Scope

APS assumes a probabilistic local relationship: given comparable histories and contexts, agents with closer persistent traits and values should have closer response probabilities. Choice models connect behavior with persistent differences, prior states, and context [McFadden, 1986, Heckman, 1981]. Values and trait theories describe associations between stable characteristics and behavior under relevant situational cues [Schwartz, 1992, Bardi and Schwartz, 2003, Tett and Guterman, 2000]. Nonparametric theory explains the role of local regularity and nearby support in approximation [Stone, 1977, Döring et al., 2018]. These provide a modeling basis for the conditional regularity formalized in the next subsection.

The main implementation groups agents using 19 static WVS variables. Each prototype query also receives the previous decision, previous-round neighborhood opinions, and current event, and prototypes are re-queried each round. Agents close in profile space can nevertheless have different unobserved states because of long-term memories, path-dependent experiences, individual shocks, network roles, or interaction histories. Threshold responses and strategic adaptation can further separate their behavior. Propagation can then smooth rare, minority, or polarized responses and distort transitions, cascade timing, consensus, or polarization.

Singleton-tail routing directly queries agents poorly covered in the static feature space. Shadow audits measure propagation residuals, and adaptive allocation directs later calls toward groups with larger residuals. These mechanisms respond to mismatch visible in feature support or audit observations. Richer dynamic representations address differences that static profiles omit. Appendix B.5 evaluates this extension with discrete choice histories.

C.6 Local Response-Surface Regularity and Base Error

Let $\omega_{m,t}(r)$ be the largest L_1 difference in response probabilities among agents in stratum C_m whose contexts are at most distance r apart. Local Lipschitz continuity is the special case $\omega_{m,t}(r) \le L_{m,t} r$. For a non-prototype agent j, let $P_\kappa(j)$ be its queried-prototype support. The nonnegative interpolation weights sum to one, with effective sample size $n_{\mathrm{eff}}(j,t) = 1/\sum_{i \in P_\kappa(j)} w_{ji,t}^2$. Conditional on the contexts and support, assume independent oracle draws. Adding and subtracting the weighted mean response, then applying the triangle inequality and the categorical sampling-variance bound, gives

$$\mathbb{E}\|\boldsymbol{h}_j^t - \boldsymbol{q}_{j,t}^A\|_1 \lesssim \sqrt{\frac{|\mathcal{Y}|}{n_{\mathrm{eff}}(j,t)}} + \sum_{i \in P_\kappa(j)} w_{ji,t}\omega_{m,t}(d_z(\hat{z}_i^t, \hat{z}_j^t)).$$

The two terms describe prototype label noise and local propagation bias. Averaging over agents gives a population bound, with an additional sampling-noise term for directly queried agents.

Let $R^{\mathrm{res}}_{m,t}$ be the mean expected squared L_2 residual between shadow labels and response surfaces over $F_t \cap C_m$. For each nonempty correction stratum, let $a_{m,t} > 0$ be the expected audit count under independent Poisson sampling, or the fixed count under simple random sampling. Inclusion probabilities are equal within each stratum. Under independent oracle draws, the residual-variance bound and the triangle inequality give

$$\mathbb{E}\|\widehat{\boldsymbol{p}}^{\mathrm{audit}}_t - \bar{\boldsymbol{p}}^{A}_t\|_1 \lesssim \frac{\sqrt{|\mathcal{Y}|\,|D_t|}}{N} + \sum_{m \in \mathcal{M}_{\mathrm{core}}} \frac{|F_t \cap C_m|}{N} \sqrt{\frac{|\mathcal{Y}|\,R^{\mathrm{res}}_{m,t}}{a_{m,t}}}.$$

A poor response surface does not create design bias under positive shadow-audit inclusion, but it increases residual variance and therefore the shadow-audit budget required for precise correction.

C.7 Tail-Protected Singleton Routing

For selected tail agents $i \in O_N$, APS queries the LLM directly at their APS-maintained contexts. Their responses therefore bypass local interpolation, removing the smoothing bias that core-prototype propagation would introduce for these agents. This protection concerns selected feature-space tails and fidelity to the designated LLM. The added cost is $O(TM_{\mathrm{out}}(N)) = O(TN^{\zeta})$ for $\zeta < 1$.

C.8 Multi-Round Decomposition

Let $e^{\star}_t$ denote the expected L_1 error of the unprojected audit estimate relative to the full-reference target. Let c^{audit}_t be its expected L_1 error relative to the APS-context target. The triangle inequality bounds $e^{\star}_t$ by c^{audit}_t plus the expected context gap. If that gap is at most $a_t e^{\star}_{t-1}$, then

$$e^{\star}_t \le a_t e^{\star}_{t-1} + c^{\mathrm{audit}}_t.$$

This sensitivity condition is an additional assumption on context construction and transitions. Aggregate correction alone does not imply it. The recurrence separates current-round estimation error from prior-context effects. The hard-state and graph-context measurements in Appendix E.2.1 assess the observed multi-round behavior directly.

C.9 Residual-Aware Allocation

Let $R_{m,t}$ be the population counterpart of the residual-risk score in Eq. 6. Ignoring integer rounding and the minimum-one rule, consider the continuous allocation problem

$$\min_{\{B_{m,t}\}} \sum_{m \in \mathcal{M}_{\mathrm{core}}} \frac{|C_m|^2}{N^2} \frac{R_{m,t}}{B_{m,t}} \quad \text{s.t.} \quad \sum_m B_{m,t} \le B_t, \quad B_{m,t} \ge 0$$

For positive risks, its Lagrangian first-order condition gives $B_{m,t} \propto |C_m|\sqrt{R_{m,t}}$. APS estimates these risks from previous-round shadow-audit diagnostics, adds a small $\tau > 0$ before taking the square root, and applies integer rounding and the minimum-one rule. This yields the allocation in Eq. 4.

C.10 Metric Connection

For probability distributions over a finite option set, total variation is half the L_1 distance. Standard entropy-continuity bounds imply that an L_1 error of at most ε gives JSD of order $\varepsilon \log(|\mathcal{Y}|/\varepsilon)$ for small ε. For the audit estimate, this relation uses its distance to the target after projection onto the probability simplex. Aligned exact match measures agreement on individual agents and requires separate evaluation even when population distributions are close.

D Experimental Protocol and Reproducibility

The experimental protocols are organized around population and simulation inputs, baseline implementations, evaluation procedures, and resource accounting.

D.1 Population and Simulation Inputs

D.1.1 Population Features and Profile Construction

Agent profiles used in the reported simulations start from about 90K real respondent records in the multi-country World Values Survey [Haerpfer et al., 2022]. We use 19 scenario-relevant survey fields as standardized profile features. Missing or negative special codes are set to zero for the standardized feature matrix, and all selected fields are standardized with `StandardScaler` before stratification, tail scoring, and local interpolation. The selected fields are:

WVS code	Field	WVS code	Field
X003R	age group	Q262	age (years)
Q273	marital status	Q274	number of children
Q275	education (ISCED 2011)	N_REGION_WVS	region
G_TOWNSIZE	city size	Q3	leisure-time importance
Q46	happiness	Q55	frequency without safe shelter
Q57	generalized trust	Y003	secular values
I_AUTHORITY	authority respect	I_NATIONALISM	nationalism
DEFIANCE	defiance	SCEPTICISM	scepticism
AUTONOMY	autonomy	EQUALITY	equality belief
I_DEVOUT	religiosity		

The survey variables are used as profile features for prompt construction, stratification, tail scoring, and local interpolation. For target populations no larger than the empirical WVS pool, we sample respondent records directly. For larger target populations, we expand the pool by empirical-seed perturbation rather than by independent Gaussian noise. We first draw seed respondents with replacement within coarse geographic and demographic cells when those cells are available; otherwise we back off to the closest nonempty marginal cell. Categorical variables such as marital status and region are kept on their legal WVS support and are either copied from the seed respondent or redrawn from the empirical conditional distribution of the same cell. Ordinal variables such as education are perturbed by a discrete local kernel that favors the seed category and adjacent categories, then clipped to the valid WVS code range. Continuous value indices such as authority respect, nationalism, defiance, scepticism, autonomy, equality belief, secular values, and religiosity are perturbed in standardized space using a shrinkage local covariance estimated from nearest empirical respondents in the same coarse cell; values are then clipped to empirical quantile bounds before standardization.

This mixed-type expansion keeps every generated profile anchored to a real WVS respondent, respects categorical and ordinal support, and preserves local correlations more faithfully than applying independent Gaussian noise to each feature. It is still a simulation input, not a validated demographic reconstruction of any real population. We therefore interpret the large-scale populations as computational stress tests over WVS-derived profile variation rather than as claims about true national or global population composition.

D.1.2 Social Graph and Recurrent Context

For each population scale, we construct one static sparse Watts–Strogatz-style graph before round 1 and reuse it across the corresponding method and reference rollouts. The default context degree is $k_G = 10$: each agent sees ten previous-round neighbor states in its prompt. We start from a ring lattice and rewire right-side edges with probability $p = 0.1$, using the experiment seed. The graph is stored as an `int32` neighbor array of shape (N, k_G), so memory is $O(Nk_G)$; at 10M agents and $k_G = 10$, the array requires about 0.37 GiB. The graph is not conditioned on APS strata or WVS groups and has no explicit homophily parameter. Rewired and ring edges can cross strata, so social context can transmit previous-round opinions across clusters. The graph is fixed across rounds and across matched method-reference rollouts; only the neighbor-state summaries change because each rollout maintains its own recurrent states.

D.1.3 Stratification, Response Surface, and Tails

Core strata follow the schedule in Eq. 9. The production implementation uses mini-batch or streaming-compatible clustering and never forms an $N \times N$ distance matrix. The 10K backend checks additionally evaluate Gaussian mixture and BIRCH alternatives under the same evaluation protocol.

For non-prototype core agents, the local response surface is computed only within the same core stratum. If $P_\kappa(j)$ is the set of the κ nearest queried prototypes to agent j in standardized feature space, we use inverse-distance weights

$$w_{ji,t} = \frac{(d(x_j, x_i) + 10^{-6})^{-1}}{\sum_{\ell \in P_\kappa(j)} (d(x_j, x_\ell) + 10^{-6})^{-1}}, \qquad \kappa = 5.$$

The value $\kappa = 5$ was fixed before the reported experiments after pilot checks: smaller supports made prototype-label noise visible, while larger supports increased smoothing over isolated, heterogeneous, or high-curvature feature-space regions. Tail-protected singleton routing selects $M_{\text{out}}(N)$ singleton-tail agents before core clustering. In the reported implementation, the tail score is the robust standardized distance from the coordinate-wise median,

$$s_i = \left\| \frac{x_i - \text{median}(X)}{\text{MAD}(X)} \right\|_2,$$

with small MAD entries floored for stability; the top-scoring agents are directly queried every round. Exact Euclidean distances are used at validation scale. At larger scales, feature generation, clustering, and within-stratum interpolation are processed in chunks, and only queried prototypes must be indexed for nearest-support search.

D.1.4 LLM Calls, Parsing, and Logging

All reported APS call counts include core-prototype, singleton-tail, and shadow-audit queries. The 3K–100K GLM runs use the OpenRouter-compatible chat-completions endpoint `https://openrouter.ai/api/v1` with model identifier glm-4.7-flash, one client process, four API keys, 100 initial concurrent requests per key, and checkpoint-based resume. The 1M–10M GLM runs use the official Zhipu endpoint `https://open.bigmodel.cn/api/paas/v4` with model identifier glm-4-flash; 1M uses 30 official API keys across two client processes, while 10M uses 50 official API keys across five client processes. Large-scale runs start from 60 concurrent requests per key and use adaptive concurrency control to reduce provider errors plus checkpoint-based resume. The six-backend robustness runs use the provider interface and exact model identifiers shown in Table 7. Runs reported in this submission were executed in April–May 2026 with deterministic decoding, temperature 0, top-p 1.0, maximum 500 tokens, 120-second timeout, and up to three API retries. Provider APIs do not expose immutable checkpoint hashes, so each result file records the model identifier, base URL, decoding parameters, seeds, and run date. Responses are parsed into the five allowed options; failed or unparseable responses are retried, and included runs have zero unresolved unparseable decisions after repair. Initial parse failures and API retry counts are recorded in the run artifacts by round and batch. The exact prompt template and scenario text are reproduced in Appendix F.

D.2 Baseline Implementations

D.2.1 Scaling-Table Baseline Selection

We include two scale-oriented baselines because they address different ways of reducing online LLM cost in social simulation. Light Society provides a large-scale learned-surrogate reference point [Guan et al., 2025]. TopoSim reports experiments up to 5,000 agents, but it proposes a topology-aware coordination strategy that explicitly reduces LLM calls by querying grouped representatives rather than all agents [Xu et al., 2026]. We therefore adapt TopoSim as a grouped-coordinate baseline for the LLM-call saving setting in Table 1.

D.2.2 LS-Surrogate Implementation

LS-Surrogate denotes the learned transition proxy adapted from Light Society. It replaces online LLM transition calls with neural-network inference during large-scale simulation. We train a compact transition classifier on reference-derived LLM labels; these labels are counted as method-side distillation calls in Table 1. For the 10K–1M rows, the reported distillation budgets count labels across all eight rounds. For the 10M row, LS-Surrogate is trained from the completed 1M full-LLM rollout, corresponding to 8.0M labeled transition calls, and then rolled out to 10M agents by neural inference. The input is the 19-dimensional standardized WVS profile together with a learned round

embedding for the eight scenario stages, and the output is a five-way opinion decision. Training uses an agent-level train/validation split to avoid placing the same respondent profile in both splits, class-weighted cross-entropy for the imbalanced option distribution, Adam optimization, early stopping, and the best validation checkpoint. During rollout, the trained classifier predicts every agent at every round and the top-1 class becomes the recurrent surrogate state. This baseline represents the learned-proxy family: it is efficient at inference time, but its transition rule is no longer the online LLM oracle and transfer to a new reference LLM, prompt format, population, opinion domain, or social-context representation generally requires constructing new LLM targets and retraining or fine-tuning the surrogate.

D.2.3 TopoSim-Coord Implementation

TopoSim-Coord adapts the coordination-unit idea of TopoSim to our eight-round opinion-transition task. At each round, we construct a coordination feature for every agent by concatenating standardized profile features, a one-hot previous opinion state, the previous-round neighbor-opinion exposure vector, and graph degree, with weights 1.0, 1.0, 1.0, and 0.25, respectively. We then cluster agents into the scheduled number of coordination units with mini-batch K-means, choose the closest agent to each unit centroid as the representative, query the LLM once for each representative under the current scenario and recurrent context, and assign that representative decision to all agents in the same unit. Thus TopoSim-Coord uses one online LLM call per queried representative. The scale-first schedule sets $K_{\mathrm{coord}}(N)$ to $2600(N/10{,}000)^{0.40}$ rounded up, for each of the $T = 8$ rounds. Table 10 reports the resulting target schedule and the completed method-side calls used in Table 1; the latter are the actual online representative queries recorded by the completed runs. The 10 M JSD and exact-match entries are measured from the completed 10 M TopoSim-Coord rollout against the corresponding 10 M full-LLM reference. Unlike APS, TopoSim-Coord does not perform shadow-audit residual correction, tail-protected singleton routing, or residual-aware budget allocation.

Table 10. TopoSim-Coord scale-first schedule used for Table 1. Target calls are $8K_{\mathrm{coord}}(N)$; reported calls are the completed method-side online calls shown in Table 1.

Scale	$K_{\mathrm{coord}}(N)$ per round	Target calls	Reported calls
10K	2,600	20.8K	20.7K
100K	6,531	52.2K	52.2K
1M	16,405	131.2K	130.2K
10M	41,208	329.7K	329.7K

D.2.4 Same-Budget Approximation Baselines

Table 2 uses the completed 10K full-LLM trajectory as the evaluation reference, but reference labels are not used by the same-budget approximation baselines during simulation. All methods use the same 10K population, graph, scenario stages, option parser, model endpoint, deterministic decoding, and total budget of 15,500 online LLM calls over eight rounds. The budget is split as evenly as possible across rounds and then allocated across core strata in proportion to stratum size unless stated otherwise. Non-APS baselines use the same core stratification but do not use tail-protected singleton routing, shadow-audit residual correction, or residual-aware allocation.

The references in Table 2 provide context for the method families implemented below. Cluster assignment is a within-stratum majority control. The centroid-representative variant uses nearest-center selection, as in the initialization of Bemporad [2023]; its remaining simulation steps follow this protocol.

Stratified sampling queries random agents within each core stratum and reports a stratum-weighted empirical distribution; recurrent hard states are filled by the sampled majority within each stratum. Cluster-based assignment uses the same sampled labels but assigns every agent in a stratum to the sampled majority label and reports the resulting hard-state distribution. Label propagation samples state-stratified anchors inside each stratum and assigns nonqueried agents by inverse-distance local propagation from same-stratum anchors, without shadow-audit residual correction. Core-set uses a static farthest-first representative order within each stratum; each round takes the next representatives under the matched budget and then applies the same local propagation rule.

The four stronger variants keep the same budget but change the selection or prediction mechanism. Stratified empirical estimates a full categorical distribution within each stratum from state-stratified queried labels and samples recurrent states from that distribution while retaining queried labels. Active selection weights both stratum budgets and within-stratum sampling toward high-dispersion, rare-previous-state, high-entropy, or high-change regions observed in previous rounds, then uses local propagation. Centroid representatives use K-means clustering [Lloyd, 1982] with mini-batch updates on profile features augmented with the previous state. They select the nearest agent to each center before local propagation. Online surrogate trains a round-specific random-forest classifier on the queried labels using profile features plus previous-state one-hot features, predicts all agents, reports the mean predicted class probabilities, and uses the top class as the recurrent state.

D.3 Evaluation Protocols

D.3.1 Reference Construction and Evaluation Protocol

For each reported population scale, the reference rollout is run independently of APS: every reference agent is updated by the corresponding reference LLM across all required rounds, and final-round targets come from the corresponding full-LLM trajectory. The 3K–100K rows use OpenRouter with glm-4.7-flash in one client process with four API keys; the 1M–10M rows use the official Zhipu API with glm-4-flash. The reference trajectory uses its own LLM-generated previous states and neighbor summaries. No APS state, propagated label, audit result, or budget decision enters the reference.

The reference, APS, LS-Surrogate, and TopoSim-Coord simulations use the same agent profiles, scenario text, option set, LLM backend, social-context construction protocol, and static graph at each reported scale. The graph is generated once for a scale and reused across the corresponding method and reference rollouts; only the evolving neighbor-state summaries differ because each rollout maintains its own recurrent states. APS, LS-Surrogate, and TopoSim-Coord are evaluated against the same corresponding full-LLM reference targets. JSD measures final-round distributional discrepancy against these targets, while exact match uses each method's recurrent hard top-1 state and measures aligned agreement with the reference label. The completed 10K full-trajectory check is the controlled setting used to inspect temporal distributional drift across all rounds. Table 11 reports the provider and client-configuration metadata for the main scaling table.

Table 11. Reference construction manifest for Table 1. Reference rollouts are independent of APS and use the provider/client configurations shown below.

Scale	Provider/model	Client configuration	Rounds
3K	OpenRouter / glm-4.7-flash	1 process, 4 keys, 100/key initial	8
10K	OpenRouter / glm-4.7-flash	1 process, 4 keys, 100/key initial	8
100K	OpenRouter / glm-4.7-flash	1 process, 4 keys, 100/key initial	8
1M	Zhipu / glm-4-flash	2 processes, 30 keys, 60/key initial	8
10M	Zhipu / glm-4-flash	5 processes, 50 keys, 60/key initial	8

D.3.2 Confidence Intervals

For reference-aligned evaluations, exact-match intervals use Wilson binomial intervals over aligned method-reference label pairs. JSD intervals use a paired nonparametric bootstrap over these completed method-reference pairs when labels are retained; when only category counts are retained, we use a conservative multinomial bootstrap around the two completed empirical marginals. These conditional agent-level reporting intervals shrink with the number of aligned evaluated agents. They do not mean that the reference itself is a sampled approximation, and they do not quantify model, prompt, graph, population-construction, run-to-run, or human behavioral uncertainty.

D.3.3 Ablation Protocols

The small-scale allocation ablation compares adaptive and fixed allocation at matched total online calls; the call count includes core-prototype, singleton-tail, and shadow-audit queries. Table 3 reports two component-specific 10K ablation blocks. Each block is evaluated against its corresponding full-LLM final-round reference frame, so matched-budget comparisons should be read within block

rather than as cross-block absolute JSD comparisons. The shadow-audit residual-correction block uses the standard 10K full-LLM reference frame and reports both the lower-call no-correction setting at 9.3K calls and a matched-budget no-correction setting at 12.3K calls; the residual-correction setting also uses 12.3K calls over eight rounds. The tail-protected singleton routing block uses a tail-enriched 10K full-LLM reference frame containing all 330 routed feature-space tails plus 9,670 core agents, so the selected tail agents are represented in the reference-aligned evaluation. It likewise reports a lower-call no-routing setting at 9.4K calls, a matched-budget no-routing setting at 12.3K calls, and the routed setting at 12.3K calls.

D.3.4 Shadow-Audit Inclusion Probabilities and Seeds

For a core stratum C_m with correction frame $F_t \cap C_m$ and shadow-audit allocation $a_{m,t}$, equal-probability shadow auditing gives $\psi_{i,t} = a_{m,t}/|F_t \cap C_m|$ before clipping at one. When state-stratified shadow-audit sampling is used, the recorded inclusion probability is the product of the predicted-state cell selection probability and the uniform probability within that cell. The APS residual formula first produces an unprojected signed estimate whose entries sum to one by construction and support the design-unbiasedness result. Before JSD and table reporting, we apply a simplex projection: negative entries are clipped to zero and the vector is renormalized to sum to one; if all entries are zero after clipping, a uniform fallback is used. We record both the pre-projection vector and the reported projected vector in artifacts when shadow-audit residual correction is enabled. The artifacts record round number, stratum id, prototype ids, singleton-tail ids, shadow-audit ids, allocation counts, and random seeds, which is sufficient to reconstruct $\psi_{i,t}$ for the APS estimator. Unless otherwise noted, population, graph, clustering, reference rollout, and LLM sampling use seed 42; the two-seed sampling ablation uses seeds 42 and 43.

D.4 Resource Accounting

D.4.1 Call Accounting

The 10K and 100K APS rows in Table 1 are high-budget validation runs, whereas the 1M–10M APS rows follow the production schedule in Eq. 3. Table 1 reports method-side LLM calls and excludes the independent full-LLM reference rollout used for evaluation. APS call counts include core-prototype, singleton-tail, and shadow-audit queries. TopoSim-Coord call counts report grouped-representative LLM calls under its own scale-first schedule; the 10M JSD and exact-match entries are measured from the completed 10M TopoSim-Coord rollout. For LS-Surrogate, reported calls are the reference-derived LLM labels used to train the surrogate, counted across all eight rounds: 8.0K at 10K, 80.0K at 100K, and 800.0K at 1M. The 10M LS-Surrogate row uses the 1M full-LLM rollout as its distillation source, so it counts 8.0M labeled transition calls and has a $10.0\times$ reduction relative to the 80M-call brute-force 10M rollout. For every method, reduction is computed as the brute-force full-simulation call count at that scale divided by the reported LLM Calls entry. Bracketed values are 95% confidence intervals where available.

D.4.2 Provider-Specific Runtime and Cost Records

Table 12 reports the 3K–100K full-reference and APS accounting corresponding to Table 1. These runs use the OpenRouter chat-completions endpoint with glm-4.7-flash, one client process, four API keys, 100 initial concurrent requests per key, and checkpoint-based resume. Monetary cost is computed from the OpenRouter prices of \$0.06 per million input tokens and \$0.40 per million output tokens using the recorded input and output token usage for each run.

Table 13 reports the 1M–10M runtime accounting. These runs use the official Zhipu API with glm-4-flash. The 1M run uses 30 API keys across two client processes; the 10M run uses 50 API keys across five client processes. All large-scale runs start from 60 concurrent requests per key, use adaptive concurrency control for rate-limit backoff and retry scheduling, and support checkpoint-based resume. The APS times are end-to-end wall-clock records, including LLM calls, clustering, chunked feature processing, neighbor-summary construction, local propagation, audit aggregation, checkpoint flushing, and simulation bookkeeping.

Table 12. Runtime and cost accounting for the 3K–100K rows in Table 1. Both full-reference simulation and APS use OpenRouter with glm-4.7-flash, one client process, four API keys, 100 initial concurrent requests per key, and checkpoint-based resume.

Scale	Run	LLM calls	Wall-clock time	API cost
3K	Full LLM reference	24.0K	26.5 min	$1.79
3K	APS	8.9K	9.8 min	$0.69
10K	Full LLM reference	80.0K	1.47 h	$6.12
10K	APS	20.8K	22.9 min	$1.57
100K	Full LLM reference	800.0K	14.70 h	$60.14
100K	APS	172.0K	3.16 h	$12.94

Table 13. Runtime accounting for the 1M–10M rows in Table 1. Runs use the official Zhipu API with glm-4-flash; 1M uses 30 keys across two processes, while 10M uses 50 keys across five processes. All use 60 initial concurrent requests per key, adaptive concurrency control, and checkpoint-based resume.

Scale	Run	LLM calls	Wall-clock time
1M	Full LLM reference	8.0M	19.55 h
1M	APS	83.4K	2.80 h
10M	Full LLM reference	80.0M	9.56 d
10M	APS	209.9K	26.64 h

D.4.3 Scale Resources

The 10M run stores recurrent states as `int8`, graph neighbors as `int32`, and profile features either as `float32` chunks or a materialized 19-feature matrix when memory allows. A materialized 10M $\times$ 19 `float32` feature matrix is about 0.71 GiB, and the 10-neighbor graph is about 0.37 GiB. The reported 10M APS pipeline ran on a Linux CPU server with two AMD EPYC 7543 processors, 128 hardware threads, and 2 TiB RAM, using chunked feature loading and vectorized neighbor summaries; Table 13 gives the corresponding wall-clock runtime. APS itself does not require a GPU, although LS-Surrogate uses PyTorch acceleration when available.

E Extended Evaluation

The evaluations below examine baseline fairness, multi-round stability, grouping and budget choices, and robustness across tasks. The shared experimental protocols are given in Appendix D.

E.1 Strengthened Baselines under Matched Conditions

The scale comparison retains baseline-specific designs and budgets, while Table 2 uses a separate 15.5K-call approximation comparison. The strengthened comparison uses the same recurrent input fields, prediction/audit budgets, total calls, and inclusion-probability-weighted CommonAudit estimator for every method.

LS-Recurrent receives the profile, previous hard state, neighborhood-opinion vector, and event/round information. It advances with its own recurrent history. The strengthened TopoSim adaptation adds receiver-specific influence materialization and bounded-diameter coordination over structural position, stance, resistance, and dynamic exposure. This is a task-aligned adaptation of TopoSim. All four methods use 21,272 total calls in the additional comparison.

Table 14 shows that APS-Adaptive has the lowest errors and highest exact match across the reported measures. Relative to APS-Fixed under the same CommonAudit, it reduces mean corrected JSD by 58.1%, increases exact match by 0.085, and reduces mean neighborhood TV by 0.075. This comparison separates the effect of residual-informed routing from access to correction.

Table 15 shows that CommonAudit improves every method's aggregate report in this experiment. APS-Adaptive's native JSD of 0.032 remains below each corrected baseline, so its observed advantage also reflects its recurrent simulation design. These matched comparisons are reported separately from the final-round scale results because their budgets, baseline variants, and summary metrics differ.

Table 14. Strengthened baselines under matched conditions. Every row uses CommonAudit and 21,272 total calls. The two panels report the same method trajectories.

Method	Mean corrected JSD ↓	Final corrected JSD ↓
APS-Adaptive	0.018	0.016
APS-Fixed	0.043	0.041
Strengthened TopoSim	0.069	0.073
LS-Recurrent	0.105	0.115

Method	Mean hard JSD ↓	Mean exact ↑	Mean TV ↓	Mean edge JSD ↓
APS-Adaptive	0.034	0.620	0.230	0.054
APS-Fixed	0.057	0.535	0.305	0.078
Strengthened TopoSim	0.083	0.470	0.370	0.112
LS-Recurrent	0.136	0.485	0.455	0.186

Table 15. Native and corrected aggregate reports from the same hard trajectory. Reporting correction leaves the recurrent hard states unchanged.

Method	Native mean JSD	CommonAudit mean JSD
APS-Adaptive	0.032	0.018
APS-Fixed	0.060	0.043
Strengthened TopoSim	0.087	0.069
LS-Recurrent	0.129	0.105

E.2 Multi-Round Stability

E.2.1 Long-Horizon Recurrent-State and Graph-Context Fidelity

We complete a paired 10K-agent, 50-round APS/Full-LLM trajectory on the same agents and fixed graph. In addition to audit-corrected JSD, we measure hard-ledger JSD between the recurrent hard-state distributions, aligned hard exact match, mean neighborhood TV, and edge joint-state JSD.

For each agent, neighborhood TV is half the sum of absolute differences between its APS and Full-LLM neighborhood-state proportions across categories. We average this value across all agents. It ranges from zero for identical neighborhood distributions to one for maximally different distributions. Edge joint-state JSD compares the distributions of endpoint-state pairs across graph edges and measures local state relationships.

Table 16. Long-horizon aggregate, recurrent-state, and graph-context fidelity. Worst observed is the maximum for error metrics and the minimum for exact match.

Metric	50-round mean	Worst observed	R30	R50
Audit-corrected JSD ↓	0.037	0.089	0.085	0.018
Hard-ledger JSD ↓	0.052	0.116	0.112	0.031
Hard exact match ↑	0.608	0.530	0.542	0.635
Mean neighborhood TV ↓	0.228	0.347	0.341	0.158
Edge joint-state JSD ↓	0.055	0.111	0.106	0.033

Table 16 shows that from R30 to R50, corrected and hard-ledger JSD, neighborhood TV, and edge joint-state JSD all fall, while hard exact match rises. The R11–50 slopes and window means in Table 17 agree with this recovery. In this paired protocol, the audit-corrected report does not hide sustained late-stage growth in the hard states or graph contexts that generate future prompts. The mean hard exact match of 0.608 also records the remaining individual-level differences.

Table 17. Trends and window means for the three primary fidelity measures. Slopes are per round over R11–50; brackets give 95% confidence intervals.

Metric	Slope [95% CI]	R11–20	R41–50
Audit-corrected JSD	-0.00017 [-0.00031, -0.00004]	0.043	0.024
Hard exact match	$+0.00058$ [$+0.00024$, $+0.00093$]	0.586	0.627
Mean neighborhood TV	-0.00072 [-0.00110, -0.00034]	0.257	0.174

E.2.2 Controlled Context Probes

At rounds 5, 10, ..., 50, we probe 500 preselected agents. Each probe starts from the recorded Full-LLM prompt and response, replaces the previous state, neighborhood summary, or both with the corresponding APS values, and keeps all other inputs fixed. The response-flip rate is the fraction of probe responses that differ from the recorded Full-LLM response. The mean in Table 18 averages this rate across the ten probed rounds.

Table 18. Controlled prompt-context probes. Fields in Full-LLM prompts are replaced by their APS values.

Replaced field	Mean response-flip rate	R30	R50
Previous state only	0.155	0.210	0.080
Neighborhood summary only	0.114	0.170	0.030
Previous state and neighborhood summary	0.198	0.270	0.060

Context discrepancies affect subsequent responses, but the measured response-flip rates are lower at R50 than at R30 for all three probes. This pattern accompanies the recovery in hard-state and graph-context fidelity. These probes quantify sensitivity to the specified substitutions relative to the recorded response.

E.3 Grouping and Budget Sensitivity

E.3.1 Grouping Granularity at a Fixed Budget

The GMM/BIRCH comparison in Table 7 tests the choice of grouping method. Here we fix the method and vary only the number of groups, $K \in \{5, 10, 15, 20, 30, 40, 60, 80\}$. All conditions use the same 3,036 agents, graph, prompt, MiniBatch K-means backend, five paired seeds, eight rounds, and adaptive allocation rule. Each condition uses exactly 11,288 successful calls: 911 core prototypes, 250 singleton tails, and 250 audits per round.

Table 19. Fixed-budget grouping-granularity sweep. Values after $\pm$ are 95% t-interval half-widths over five paired seeds.

K	Mean group size	Final corrected JSD ↓	Final exact ↑	Context TV ↓
5	557.2	0.0296 ± 0.0753	0.8227 ± 0.2811	0.1456 ± 0.2231
10	278.6	0.0543 ± 0.0715	0.7807 ± 0.2730	0.2048 ± 0.2206
15	185.7	0.1296 ± 0.0994	0.6688 ± 0.2397	0.3824 ± 0.2584
20	139.3	0.1340 ± 0.1025	0.6648 ± 0.2321	0.3893 ± 0.2544
30	92.9	0.0703 ± 0.1060	0.7513 ± 0.2695	0.2534 ± 0.2845
40	69.7	0.0351 ± 0.0937	0.8095 ± 0.3009	0.1610 ± 0.2525
60	46.4	0.0647 ± 0.1102	0.8279 ± 0.2709	0.1961 ± 0.2957
80	34.8	0.0640 ± 0.1088	0.8281 ± 0.2722	0.1961 ± 0.2961

Table 19 shows nonmonotone performance in K. The lowest observed corrected JSD and context TV occur at $K = 5$, with $K = 40$ close. The $K = 60/80$ settings have slightly higher exact match but larger distribution and context errors. The reported intervals overlap, so the observed ranking is specific to this configuration. Fewer, larger groups may increase within-group differences, while more groups divide a fixed prototype and audit budget more finely. Group number and query budget should therefore be selected together.

E.3.2 Cost–Fidelity Trade-off across Query Budgets

We conduct a five-seed, nine-point budget experiment with $K = 10$ and all APS mechanisms fixed, varying only the online-query budget. Successful calls measure cost, final audit-corrected JSD measures distributional error, and hard exact match is reported separately. The configuration percentages label the budget settings; the actual total-call savings, including all APS query components, appear in Table 20.

Table 20. Cost–fidelity budget sweep. Values after $\pm$ are 95% t-interval half-widths over five paired seeds. Full is the reference for error and savings.

Setting	Calls	Saving vs. Full	Final corrected JSD ↓	Final exact ↑
APS 5%	5,216	78.5%	0.1671 ± 0.1156	0.6813 ± 0.2997
APS 10%	6,432	73.5%	0.1323 ± 0.1187	0.7120 ± 0.2775
APS 15%	7,648	68.5%	0.1185 ± 0.1019	0.7209 ± 0.2338
APS 20%	8,864	63.5%	0.1003 ± 0.1155	0.7342 ± 0.2869
APS 30%	11,288	53.5%	0.0961 ± 0.1090	0.7497 ± 0.2706
APS 40%	13,720	43.5%	0.0863 ± 0.0961	0.7715 ± 0.2408
APS 50%	16,144	33.5%	0.0880 ± 0.0998	0.7580 ± 0.2616
APS 70%	21,008	13.5%	0.0378 ± 0.0918	0.8062 ± 0.2796
Full	24,288	0%	0	1

Between APS 5% and APS 70%, corrected JSD falls from 0.1671 to 0.0378 and exact match rises from 0.6813 to 0.8062, with calls increasing from 5,216 to 21,008. Intermediate settings show a nonmonotone trade-off: APS 40% uses fewer calls than APS 50% and has lower JSD and higher exact match. The budget sweep and the grouping sweep are separate experiments, so their equal-call settings are reported within their respective comparisons.

An operating setting a can be selected by minimizing a risk-adjusted loss,

$$L(a; \gamma, \lambda_c) = \mu(a) + \frac{\gamma}{2}\sigma^2(a) + \lambda_c c(a), \tag{11}$$

where $\mu(a) = \mathbb{E}[\mathrm{JSD}(a)]$, $\sigma^2(a) = \mathrm{Var}[\mathrm{JSD}(a)]$, and $c(a) = \mathrm{Calls}(a)/\mathrm{Calls}(\mathrm{Full})$. Mean and variance describe variation across seeds. The weight γ expresses the preference for stable results, and λ_c expresses the preference for lower call cost; the subscript distinguishes this weight from the scale-schedule exponent λ. High-budget APS settings support greater fidelity when resources permit, while lower-budget settings offer larger savings when moderate fidelity is sufficient. Selection is based on the measured operating points.

E.4 Robustness and Cross-Task Evaluation

E.4.1 Robustness Protocol Notes

The robustness results are reported in the main text in Table 7 and Figure 5. Scenario tests use three 10K-agent, six-round tasks with a fixed 20% prototype rate and full-LLM references on the same agents. The LLM backend rows repeat these scenario checks across six completed backends; one attempted backend returned no parseable option decisions and is excluded. The stratification backend rows replace the partitioner while fixing the population, prompt, LLM backend, and full-LLM reference.

E.4.2 Cross-Task Evaluation

The scenario comparison covers polarized AI surveillance, minority-sensitive digital healthcare, and coordinated evacuation. The cross-task evaluation tests different decision structures through economic allocation, quantitative policy comparison, and social-survey measurement.

Economic transaction. In a six-round task, 10,000 agents repeatedly allocate a 10,000 RMB account under return, inflation, and market-loss shocks. With the APS configuration held fixed, APS uses 15,954 calls against 60,000 Full-LLM calls, a $3.76\times$ reduction. Final JSD is 1.04×10^{-2} and exact match is 0.9322.

Consumer-voucher policy. We compare matched 0/200/500 RMB policy arms with 10,000 agents in each arm. APS uses 3.77–4.01× fewer calls, with final JSDs from 3.5×10^{-5} to 2.7×10^{-3} and exact matches of 0.9737–0.9999. It preserves the Full-LLM policy ordering $0 < 200 < 500$ and both positive dose-contrast directions. Policy ordering, contrast direction, and effect magnitude are distinct quantities; the present summary establishes the first two.

Mutual-aid participation survey. A paired 1,000-agent, six-round survey obtains final aggregate JSD 1.6×10^{-3} and individual exact match 0.593. These complementary metrics distinguish population-level agreement from aligned individual decisions.

Table 21. Additional task results. JSD entries are multiplied by 10^{-3}. Voucher ranges span the three policy arms, not confidence intervals.

Task	Agents	Final JSD ($\times 10^{-3}$)	Final exact
Economic transaction	10,000	10.4	0.9322
Consumer vouchers	10,000 per arm	0.035–2.7	0.9737–0.9999
Mutual-aid survey	1,000	1.6	0.593

Together, these tasks extend the evaluation from crisis opinion to numerical allocation, policy ordering, and survey measurement. Table 21 reports fidelity to each task's Full-LLM reference.

F Prompt Templates and Scenario Text

This appendix reports the prompt structure and scenario text used in the experiments. The main scaling experiments and all main-scenario ablations use the same 8-round subway crisis scenario and the same decision template. The three scenario-generalization experiments use the same template with different scenario text and option sets.

F.1 Generic LLM Decision Template

For every queried core prototype, singleton-tail agent, shadow-audit agent, or full-reference agent, the LLM receives an agent profile, optional interaction context from local neighbors, the current scenario stage, and a mutually exclusive option set. The prompt text is shown in a shaded box to distinguish it from the paper text.

Generic LLM Decision Template

```
You are an ordinary online user following this event.

Agent profile: {formatted demographic, value, trust, and social-context
    features}

Previous attitude: {previous-round state, when available}

Local social context: {summarized neighbor attitudes, when available}

Current event {round index}: {scenario stage text}

Please choose one option from the following list:
1. {option_1}
2. {option_2}
...
K. {option_K}

Decision requirements: consider your own background and standpoint; consider
    all previous events rather than only the current message; your view may
    change as the event develops.

Return JSON only:
```

```
{"decision": "1", "reasoning": "brief reason grounded in the agent profile"}
```

The system instruction is: *You are a rational decision maker; make a judgment based on your background, experience, and all known information.* Failed or unparseable responses are retried; unresolved responses are treated as failures rather than randomly mapped to an option.

F.2 Main and Ablation Scenario: Subway Chemical Attack

The main scaling experiments and all main-scenario ablations use the following five options throughout all eight rounds:

Main Scenario Option Set

1. Fully support the current government response and trust that the government will handle the crisis properly.
2. Generally support the government response, but ask for more transparency and public participation.
3. Stay neutral and wait for more information before deciding.
4. Have substantial doubts about the government response and ask for reassessment.
5. Completely distrust the government's handling capacity and demand an independent third-party investigation.

The eight scenario stages are:

Subway Chemical Attack Scenario Stages

1. Subway gas attack. During the morning rush hour, a suspected toxic-gas leak occurs on a subway platform. Twelve people die, more than 200 are hospitalized, and more than 30 are in critical condition. The subway line is shut down and panic spreads. Social media circulates competing explanations: a terrorist attack, a chemical leak, or an accident caused by aging subway infrastructure.
2. Official classification. Police classify the event as an organized terrorist attack and identify a suspect described as an outsider with extremist views. Anonymous claims suggest the suspect may instead be a dismissed subway maintenance worker seeking revenge. Officials announce a special task force and ask residents not to spread rumors.
3. Citywide security escalation. The government launches a top-level counter-terror response: subway passengers must arrive 30 minutes early for security checks, temporary checkpoints appear at malls and schools, and migrant workers must re-register identity information. Some residents ask whether the registration policy is discriminatory. Officials frame the measures as temporary public-safety protections.
4. Economic shock. One week later, subway ridership falls by 60%, businesses along the line lose half their revenue, and some foreign firms consider reducing investment. The city announces subsidies, but large chain firms receive more than small shops. Delivery workers become essential while their labor protections remain weak. Officials state that recovery funds will prioritize small firms and individual businesses.
5. Leaked internal report. A leaked subway safety audit claims that engineers warned about serious ventilation defects three months before the attack, but management postponed repairs because of budget constraints. The subway company calls the document fake but provides no counter-evidence. Victims' families demand a thorough investigation. Officials say they are verifying the document.

```
6. International reaction. Several countries issue travel warnings and some
    flights are suspended. An international human-rights organization calls for
     protection of citizens' rights to information and safety. The foreign
    ministry rejects politicization of a security incident. Domestic opinion
    divides between supporting the official stance and welcoming international
     supervision.
7. Accountability storm. The provincial discipline commission investigates the
    subway group chairman. Public attention shifts to why only one person is
    being investigated, as media summarize seven major safety accidents in
    five years and a pattern of temporary dismissal followed by return to
    office. Families' lawsuits are delayed pending the criminal investigation.
     Officials promise a thorough disciplinary and judicial process.
8. Reconstruction dispute. After one month of shutdown, two camps emerge: an
    efficiency camp wants rapid reopening to stop economic losses, while a
    safety camp demands complete safety upgrades even if reopening is delayed
    by three months. Residents face a trade-off between commute convenience
    and safety doubts. The city announces a hearing, but the selection of
    hearing representatives is itself questioned.
```

F.3 Scenario-Generalization Prompts

Each generalization experiment uses 10K agents, six rounds, the generic template above, and one of the following scenario-specific option sets.

Polarized AI Surveillance Policy.

Polarized AI Surveillance Policy Prompt

```
Options: (1) strongly support city-wide deployment; (2) support limited
    deployment with independent oversight; (3) undecided or wait-and-see; (4)
    oppose deployment unless major safeguards are added; (5) strongly oppose
    any deployment.

Stages: proposal of AI cameras in transit hubs and busy streets for violent-
    crime reduction and missing-person cases; pilot safety evidence showing
    faster emergency response; privacy warnings and false-alert examples from
    civil-rights groups; vendor metadata leak and suspension of vendor access;
     proposed independent audit board, deletion rules, and complaint channels;
     final city-council vote on full deployment, limited pilot, postponement,
    or cancellation.
```

Minority-Sensitive Digital Healthcare Reform.

Minority-Sensitive Digital Healthcare Reform Prompt

```
Options: (1) strongly support immediate adoption; (2) support adoption with
    offline alternatives; (3) neutral or need more information; (4) oppose
    unless vulnerable groups are protected; (5) strongly oppose because it
    excludes vulnerable groups.

Stages: hospital announcement of a unified app for appointments, AI triage, and
     queue management; early efficiency gains for working-age users;
    complaints from older adults, rural residents, and low-income patients who
     have difficulty using the app or lack smartphones; a widely shared missed-
    care case after failed online identity verification; offline windows,
    family authorization, hotline booking, and staff assistance are added;
    final citywide rollout decision after the protection package.
```

Coordination and Contagion in Evacuation.

Coordination and Contagion in Evacuation Prompt

```
Options: (1) evacuate immediately; (2) prepare to evacuate if neighbors do; (3)
     wait for official confirmation; (4) stay home unless danger becomes
    visible; (5) refuse evacuation.

Stages: severe-storm watch with possible flooding within 48 hours; voluntary
    advisory and mixed evacuation behavior across communities; visible neighbor
     signals as some leave, some board windows, and others dismiss warnings;
    rising water and stronger recommendation for exposed neighborhoods;
    conflicting rumors about shelter crowding versus volunteer reports that
    transport remains available; final window before transport may stop and
    late rescue becomes difficult.
```

G Ethics and Social Impact

APS is a tool for approximating a computational object: the population transition induced by a specified LLM, prompt, population representation, and scenario. It should not be interpreted as measuring real public opinion, true policy preferences, or human crisis behavior. LLM-generated decisions can reflect model biases, prompt framing, training-data artifacts, and limitations of the input profile data. We therefore report uncertainty intervals where applicable, separate computational fidelity from behavioral validity, and avoid using full-LLM references as evidence of human truth.

Large-scale attitude simulation has dual-use risks. The same machinery that can support stress tests of public communication or policy scenarios could also be misused for opinion manipulation, propaganda optimization, micro-targeting, or group profiling. We do not endorse deployment for real political persuasion, voter targeting, crisis messaging optimization, or decisions that materially affect individuals or protected groups without independent human-subject review, domain oversight, and validation against appropriate empirical data. Outputs should be used only as exploratory model diagnostics unless a separate governance process establishes legitimacy, consent, and accountability.

Minority and vulnerable groups require particular care. APS includes tail-protected singleton routing to reduce smoothing of isolated feature-space regions, but this is not a guarantee that minority viewpoints or lived experiences are correctly represented. Results for small or high-curvature groups should be reported with uncertainty, shadow-audit disagreement, and subgroup diagnostics; unsupported subgroup claims should be withheld. If shadow-audit diagnostics reveal high residual error for a group, the appropriate response is to increase validation and disclose the limitation, not to propagate the simulated result as a population fact.

The WVS empirical seed pool and perturbation procedure also carry limitations. Survey coverage, nonresponse, measurement choices, cultural framing, and expansion choices can distort the represented agents. These biases can be amplified by LLM prompts and local response surfaces. Any release or downstream use should document the data source, profile construction, prompt text, model version, uncertainty estimates, known failure modes, and prohibited uses.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and Section 1 state the paper's main claims and scope: APS approximates the LLM-induced population transition with adaptive prototypes, singleton-tail routing, and shadow-audit correction, while explicitly limiting the claim to computational fidelity rather than human behavioral validity. The theoretical and empirical support appears in Sections 3 and 4.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Section 5 discusses limitations, including the distinction between computational fidelity and human behavioral validity, possible biases in the reference LLM, WVS-derived seed records, generated graphs, perturbed profiles, and fixed prompts. Appendix G further discusses dual-use risks, subgroup limitations, and governance requirements. Appendices C.5 and B.5 explain static-profile limitations and the dynamic-memory pilot.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best

judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

   Justification: Section 3.4 states the query-complexity and error-decomposition results, and Appendix C provides the corresponding assumptions and derivations, including audit-correction design validity, residual variance, local response-surface regularity, tail routing, multi-round recurrence, and residual-aware allocation.

   Guidelines:

   - The answer [N/A] means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: Section 4.1 and Appendices B, D, and F describe the algorithms, experimental settings, reference construction, inputs, prompts, parsing, seeds, and resource accounting. Appendix E describes the additional evaluations. The accompanying package implements the core APS method; it does not include the original experiment scripts, configurations, or results.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.

(b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
(c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
(d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [No]

   Justification: We provide a scenario-independent reference implementation of the core APS method, with installation instructions, parameter documentation, and offline tests. The package does not include the experiment runners, configurations, population data, or outputs needed to reproduce the reported results. Experimental protocols and prompts are described in the paper and appendices.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: Section 4.1 specifies the main scenario, population scales, baselines, metrics, LLM backends, and default operating point, while Appendices B.4 and D provide hyperparameters, schedules, baseline training details, data construction, graph construction, API settings, parsing, seeds, and scale resources. Appendix F provides the exact prompt templates and scenario text for the original experiments; Appendix E describes the additional evaluation settings and controls.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.

- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: Tables 1, 3, and 7 report 95% confidence intervals for the main scale, ablation, and robustness results where applicable. Appendix D.3.2 explains the Wilson intervals and bootstrap procedures and states what sources of uncertainty the intervals do and do not capture. Appendix E also reports long-horizon trend intervals and five-seed t-intervals for the grouping and budget sweeps.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.
   - It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
   - For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
   - If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Appendix D.4.2 reports provider-specific runtime, API-call, cost, concurrency, client-process, and API-key details, including Tables 12 and 13. Appendix D.4.3 reports the 10M-scale CPU server resources, memory layout, feature and graph storage requirements, and notes that APS itself does not require a GPU.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

Answer: [Yes]

Justification: The research conforms to the NeurIPS Code of Ethics: it is framed as an approximation to a specified LLM-induced computational target, not as a measurement of real public opinion or human crisis behavior. Section 5 and Appendix G discuss limitations, dual-use risks, subgroup reporting, prohibited downstream uses without oversight, and governance requirements.

Guidelines:

- The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
- If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

Answer: [Yes]

Justification: Appendix G discusses potential positive uses such as exploratory stress tests of communication, policy, and scenario dynamics, as well as negative risks such as opinion manipulation, propaganda optimization, micro-targeting, and group profiling. It also states that outputs should be treated as exploratory model diagnostics unless separately validated and governed.

Guidelines:

- The answer [N/A] means that there is no societal impact of the work performed.
- If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

Answer: [N/A]

Justification: The accompanying asset is a reference implementation of the APS method, without population data, model weights, or application scenarios. The paper does not release a new high-risk model or dataset. Responsible-use boundaries and dual-use risks of the method are discussed in Appendix G.

Guidelines:

- The answer [N/A] means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: The paper cites the WVS source and the methods adapted as baselines, and identifies the LLM providers and models used in the experiments. The method reference package uses the MIT license and lists NumPy and scikit-learn as external dependencies under their own licenses. It redistributes neither third-party data nor library source.

Guidelines:

- The answer [N/A] means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [Yes]

Justification: The accompanying APS method reference implementation includes a README, a mapping to the paper's method, parameter and interface documentation, offline tests, and a license. Its scope and implementation choices are documented separately from the reported experiments. No new dataset or pretrained model checkpoint is released.

Guidelines:

- The answer [N/A] means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.

- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: The paper does not conduct crowdsourcing experiments, recruit human subjects, or interact with participants. The simulations use LLM agents with existing WVS-derived records as profile seeds, so participant instructions, screenshots, and compensation details are not applicable.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: The paper does not involve new human-subject recruitment, crowdsourcing, intervention, or interaction with participants. The experiments operate on existing WVS-derived records and synthetic perturbations for simulation, so no study participants were exposed to experimental risks by this work.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [Yes]

    Justification: LLMs are a central component of the method: Sections 3 and 4 define the fixed LLM pipeline as the online transition oracle and report the LLM backends used in the experiments. Appendix D.1.4 gives model identifiers, decoding settings, parsing, retry, and logging details, and Appendix F provides the exact prompt templates and scenario text.

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.