In-Context Credit Assignment via the Core

author names withheld

Under Review for NExT-Game 2026

Abstract

We propose incentive-aligned mechanisms for *in-context credit assignment*: the task of assigning credit for AI-generated content (e.g. code, news articles, short-form videos) among creators whose intellectual property appears in the context window. Our approach is based on the *least core* solution concept from cooperative game theory, which distributes value in a way that is as stable as possible by ensuring that no subset of creators is significantly under-compensated relative to the value they could generate on their own. We develop algorithms for approximating the least core, which leverage novel routines for constraint seeding and constraint separation. On a web retrieval credit assignment task, we find that our approaches are capable of approximating the least core using orders of magnitude fewer LLM calls compared to alternative methods.

1. Introduction

The creator economy is currently undergoing a massive transformation as a result of the widespread adoption of generative AI tools. Across domains such as text, images, and video, AI tools have enabled cheaper and faster production of content, in ways which have not previously been possible(e.g. [1, 20]). In many use cases, this process involves synthesizing creators' intellectual property (IP) that appears in the model's *context window* into a final user-facing content item.

For example, suppose a user queries a language model which uses a web search tool to aggregate and summarize information across many websites (e.g. code repositories, news articles) when generating its response to the query. As another example, suppose a user prompts a text-to-video model to generate a video featuring the likenesses of several individuals (e.g. celebrities or fictional characters). In each of these settings, the resulting content typically yields some sort of value for the end user and/or AI platform (measured in terms of revenue, user engagement, or otherwise). The goal of this paper is to design mechanisms that distribute credit for this value across the creators whose IP appears in-context, in a way that respects the incentives of the creators. Concretely, we aim to answer the following question:

Q: *What is an economically principled and computationally tractable way to allocate credit for AI-generated content among a set of creators whose IP appears in the model's context window?*

Our first conceptual contribution is to model the in-context credit assignment problem as a *coalitional game*, where creators work together to generate some value. In this setting, there is an unknown ground-truth reward function that determines the (counterfactual) value that would have been generated by any coalition of the creators. Rewards can be binary (e.g. "Do the aggregated web pages contain enough information to answer the user's query?") or continuous (e.g. "What is the user engagement that would have been generated if only a subset of likenesses were used in the video?"). Instead of assuming access to the ground-truth reward function, we posit access to

© .

an *estimated* reward function (e.g., via a learned reward model or LLM-as-a-Judge). Our solution concept is the *least core* [13]: the set of reward allocations that compensates every coalition of creators by minimizing the largest *reward deficit* (i.e. amount of under-compensation relative to the counterfactual value generated) across all coalitions.

We investigate the sensitivity of the core to differences between the estimated and ground-truth rewards in App. B. We show that the sensitivity of the least-core deficit can be bounded in terms of reward errors weighted by a *balanced* distribution (i.e., the marginal membership probability for each *player* is the same) over coalitions. Our result can be viewed as a quantitative analog of the *Bondareva-Shapley theorem* from cooperative game theory [33].When rewards are binary, we give a negative result: an estimated reward function with exponentially-few misclassifications can yield a high deficit even when the ground-truth deficit is zero. Nonetheless, we show that sensitivity of the deficit is bounded by the worst-case probability of misclassification over a balanced distribution.

Next we develop general-purpose algorithms for finding allocations in the estimated least core. While the problem of computing the (estimated) least core can be formulated as a linear program, the number of constraints is exponential and so it cannot be solved directly for more than a modest number of creators. As such, we develop iterative separation-oracle based techniques.In App. E.1, we show that the ellipsoid method instantiated with a sampling-based separation oracle for finding under-compensated coalitions finds, with high probability, an approximate least-core allocation with polynomial sample complexity and polynomial runtime. Practically, we implement a *constraint generation (CG)* algorithm and instantiate it with a variety of different seeding and separation routines for finding under-compensated coalitions.

We evaluate our methods on a semi-synthetic web retrieval credit assignment task, where the goal is to aggregate information across many different data sources in order to answer a set of questions. We find that our CG algorithm instantiated with LLM-based separation oracles approximates least-core allocations using orders of magnitude fewer LLM calls compared to relevant baselines.

2. Setting and background

We cast the problem of in-context content credit assignment in the language of cooperative game theory. There is a set $N = \{1, \ldots, n\}$ *creators*, where each $i \in N$ is the creator of some IP that is used in the content generation process. We model rewards via a coalitional value function $v : 2^N \to [0, 1]$. The total reward generated as a result of the content created by N is the quantity to be distributed among the creators in N and is normalized to $v(N) = 1$. Likewise, the baseline reward generated by the empty set is zero. For a coalition of creators $S \subseteq N$, $v(S) \in [0, 1]$ represents the counterfactual reward that the subset S of creators would have generated on their own. While counterfactual rewards are generated according to the ground-truth value function v, we only assume that the platform has access to some *estimate* of v, which we denote by $\widehat{v} : 2^N \to [0, 1]$ (also normalized so that $\widehat{v}(N) = 1$ and $\widehat{v}(\emptyset) = 0$; v and $\widehat{v}$ agree on the grand coalition N since that is the reward observed by the platform). This framing captures two important examples of AI content generation:

1. *LLM web search.* A user queries an LLM, which draws on several web sources when generating its answer. N is the set of webpages referenced in the LLM response. The reward $v(N)$ generated from this interaction is proportional to user engagement/satisfaction with the platform.For a coalition $S \subseteq N$, $v(S)$ denotes the counterfactual reward of the response that would have been generated if the LLM were only allowed to draw from webpages in S when generating its answer. An LLM-as-a-Judge or may be used as a value function estimate $\widehat{v}$.

2. *AI video generation.* A prompt author generates an AI video featuring the likenesses of the creators in N. The video accrues some reward $v(N)$, which may be ad revenue proportional to the amount of engagement (e.g. clicks, likes, or watch-time) it receives. For a subset of creators $S \subseteq N$, $v(S)$ is the counterfactual reward that would have been generated from a video containing only those S creators.In video generation, $\widehat{v}$ may be an internal tool used to judge video quality, learned from, e.g., the performance of other videos on the platform. (Here, $v(S)$ is not directly observable short of creating a new video and letting it accrue reward on the platform.)

Definition 1 *A value-sharing scheme is a vector* $\boldsymbol{u} = (u_1, \ldots, u_N) \in [0,1]^N$ *such that* $\sum_{i \in N} u_i = 1$*, where* $u_i \in [0,1]$ *is creator* i*'s share of the reward.*

The core is the set of value-sharing schemes such that each subset of creators $S \subseteq N$ receives total value at least $v(S)$. The core is rarely nonempty.Therefore, we consider a relaxation, the ε-core, where each subset is under-compensated by an amount at most ε compared to their value $v(S)$.

Definition 2 (ε-(estimated) core) *For* $\varepsilon \geq 0$*, the* ε*-core and* ε*-estimated core are the polytopes*

$$\mathsf{core}_\varepsilon = \left\{ \boldsymbol{u} \in [0,1]^N : \textstyle\sum_{i \in N} u_i = 1; \;\; \sum_{i \in S} u_i \geq v(S) - \varepsilon, \;\; \forall S \subseteq N \right\},$$
$$\widehat{\mathsf{core}}_\varepsilon = \left\{ \boldsymbol{u} \in [0,1]^N : \textstyle\sum_{i \in N} u_i = 1; \;\; \sum_{i \in S} u_i \geq \widehat{v}(S) - \varepsilon, \;\; \forall S \subseteq N \right\}.$$

The least core is the ε-core with the smallest value of ε for which the ε-core is nonempty.

Definition 3 (Least core) *Let* $\varepsilon^* = \min\{\varepsilon \geq 0 : \mathsf{core}_\varepsilon \neq \emptyset\}$ *and* $\widehat{\varepsilon} = \min\{\varepsilon \geq 0 : \widehat{\mathsf{core}}_\varepsilon \neq \emptyset\}$*. The least core is the set* $\mathsf{core}_{\varepsilon^*}$*. The least estimated core is the set* $\widehat{\mathsf{core}}_{\widehat{\varepsilon}}$*. The quantities* ε^* *and* $\widehat{\varepsilon}$ *solve the following linear programs.*

Least Core LP:

$$\varepsilon^* = \min_{\varepsilon, \boldsymbol{u}} \{\varepsilon : \boldsymbol{u} \in \mathsf{core}_\varepsilon\} \tag{1}$$

Least Estimated Core LP:

$$\widehat{\varepsilon} = \min_{\varepsilon, \boldsymbol{u}} \{\varepsilon : \boldsymbol{u} \in \widehat{\mathsf{core}}_\varepsilon\} \tag{2}$$

Ideally we would like to implement a value-sharing scheme $\boldsymbol{u} \in \mathsf{core}_{\varepsilon^*}$. However, since the true value function v is unobservable, allocations in $\mathsf{core}_{\varepsilon^*}$ cannot be exactly computed. As a result, we focus on computing schemes in $\widehat{\mathsf{core}}_{\widehat{\varepsilon}}$, and show that under reasonable conditions these value-sharing schemes are not too far from those in $\mathsf{core}_{\varepsilon^*}$.

The least (estimated) core is a set of value-sharing schemes. We break ties among schemes in the set by minimizing ℓ_2 norm. The resulting allocation is referred to as the *egalitarian* least core.

3. Computing the least core

Our algorithms for approximating the least core are variants of the classical ellipsoid and constraint generation (CG) methods [17, 21]. These methods solve a convex optimization problem (in our case, LP (2)) by iteratively refining a feasible region using separating hyperplanes, without explicitly enumerating all constraints. Avoiding constraint enumeration is critical for core computation, as LP (2), which has 2^N constraints, cannot be written in memory for even a moderate number of creators.

The ellipsoid method [16, 17] maintains a shrinking outer approximation (an ellipsoid) of the feasible region using separating hyperplanes. While the ellipsoid method enjoys worst-case polynomial runtime guarantees, it is inefficient to run in practice. CG is a more practical algorithm: instead of maintaining a shrinking ellipsoidal approximation to the feasible region, it progressively refines a

Initialize $\mathcal{S}^1 \subseteq 2^N$
for $t = 1, \ldots, T$ **do**
  Solve $\widehat{\mathrm{LP}}(\mathcal{S}^t)$; let ε be the returned optimal deficit
  Solve $\widehat{\mathrm{QP}}(\mathcal{S}^t, \varepsilon)$; let $\boldsymbol{u}$ be the returned value-sharing scheme
  if $\mathrm{GetUnhappyCoalition}(\boldsymbol{u}, \varepsilon) = \texttt{None}$, **return** $\boldsymbol{u}, \varepsilon$
  $\mathcal{S}^{t+1} \leftarrow \mathcal{S}^t \cup \mathrm{GetUnhappyCoalition}(\boldsymbol{u}, \varepsilon)$.
end
return $\boldsymbol{u}, \varepsilon$

Algorithm 1: Joint Constraint Generation for Egalitarian Least Core

polyhedral approximation by explicitly accumulating violated constraints. While CG does not have poly-time guarantees, it is a popular method for solving optimization problems in practice [10].

Both the ellipsoid method and CG require the existence of a *separation oracle* capable of providing a hyperplane which separates a given point from the feasible region—in our setting this is the task of finding an under-compensated coalition, if one exists. We show how to use these algorithms in our setting, and we provide several separation oracles for in-context credit assignment.

Definition 4 (Separation oracle) *A separation oracle for the ε-estimated core takes as input a candidate value-sharing scheme $\boldsymbol{u} \in [0,1]^n$ and deficit $\varepsilon \geq 0$, and returns a coalition S such that $\sum_{i \in S} u_i + \varepsilon < \widehat{v}(S)$, or returns* `None` *if no such coalition exists.*

Since there is no generic poly-time exact separation oracle for core computation, the separation oracles we provide are either approximate (via coalition sampling) or based on LLM prompting.

We now propose practical CG methods for least estimated (egalitarian) core computation. For a collection of coalitions $\mathcal{S} \subseteq 2^N$, let $\widehat{\mathrm{LP}}(\mathcal{S})$ denote the restricted least estimated core LP

$$\min_{\boldsymbol{u} \in [0,1]^n} \left\{ \varepsilon : \textstyle\sum_{i \in N} u_i = 1; \;\; \sum_{i \in S} u_i \geq \widehat{v}(S) - \varepsilon, \; \forall S \in \mathcal{S} \right\}. \tag{3}$$

Similarly, given a collection of coalitions $\mathcal{S} \subseteq 2^N$ and a candidate least deficit ε, let $\widehat{\mathrm{QP}}(\mathcal{S}, \varepsilon)$ denote the restricted egalitarian least estimated core quadratic program

$$\min_{\boldsymbol{u} \in [0,1]^n} \left\{ \|\boldsymbol{u}\|_2^2 : \textstyle\sum_{i \in N} u_i = 1; \;\; \sum_{i \in S} u_i \geq \widehat{v}(S) - \varepsilon, \; \forall S \in \mathcal{S} \right\}. \tag{4}$$

Algorithm 1 describes the joint CG algorithm that solves the least estimated core LP and the egalitarian least estimated core QP simultaneously, by growing $\mathcal{S}$ at each iteration and re-solving $\widehat{\mathrm{LP}}(\mathcal{S})$ and $\widehat{\mathrm{QP}}(\mathcal{S}, \varepsilon)$. If equipped with an *exact* separation oracle, Algorithm 1 terminates with the egalitarian least estimated core allocation. Unlike ellipsoid-based algorithms, it is generally not possible to provide (polynomial) bounds on the number of CG rounds. Nonetheless, with a sampling-based separation oracle, we prove that the returned allocation is in the least estimated core with high probability, and the returned allocation minimizes ℓ_2 norm among all feasible allocations (Thm 17).

Yan and Procaccia [41] (and the corresponding implementation available in the `pyDVL` package [36]), approximate the egalitarian least core by (1) solving the least core LP using their sampling method then (2) solving the *full* QP using the returned least core deficit. A major shortcoming of this approach is if the least core LP is too large to be written down in memory—as is the premise that necessitates Yan and Procaccia to develop sampling methods—then so is the QP.

Our algorithms may be instantiated with *any* separation oracle, not just the sampling-based method of Algorithm 2. As one example, Algorithm 3 uses an LLM to identify violated coalitions. This approach is inspired by the recent line of work on *LLM-as-a-Judge*, in which language models are used to evaluate or critique candidate outputs. In our setting, the LLM plays a related role by acting as a learned separation oracle: given a candidate allocation, it proposes coalitions that are likely to violate the core constraints, effectively serving as an adversarial tester that surfaces counterexamples. While this oracle does not admit the same probabilistic guarantees as the sampling-based method, it offers a flexible and practical heuristic for guiding the search toward informative constraints, particularly in settings where domain knowledge can be encoded through prompting.

4. Experiments

We evaluate our CG methods on a semi-synthetic webpage retrieval benchmark, using both the BrowseComp-Plus dataset [5] and a dataset of our own creation. We use Gemini 3 Flash [14] as the value function estimate (via LLM-as-a-Judge grading), and Gemini 3 Pro [15] for generating our synthetic dataset. We test both egalitarian and non-egalitarian variants of our CG algorithms and find that they substantially outperform the sampling baseline of Yan and Procaccia [41] in terms of the number of LLM calls required.

We compare three families of algorithms: random sampling (**YP**; Algorithm 5), our constraint generation approach (**CG**; Algorithm 1), and a zero-shot (**ZS**) method in which we ask the LLM to directly output a least core utility vector in a single query. We assign binary rewards $\hat{v}(S) \in \{0, 1\}$ to each coalition S by prompting the model to answer the question using only the documents in S (instructing it to ignore world knowledge), then grading the response with a second call.

Evaluation metrics. A notable difficulty is the lack of a ground-truth $(\varepsilon^*, \mathbf{u}^*)$ to compare to. Even restricting to small document sets where we could solve the full LP, intrinsic noise in LLM responses prevents exact evaluation. We therefore construct a holdout set $\mathcal{H}$ of 2^{12} coalitions sampled uniformly at random, and for each method's utility vector $\mathbf{u}$ we compute the *holdout* ε, $\hat{\varepsilon}_{\mathcal{H}}(\mathbf{u})$:

Definition 5 $\hat{\varepsilon}_{\mathcal{H}}(\mathbf{u})$ *measures the out-of-sample quality of a utility vector* $\mathbf{u}$ *against held-out coalitions* $\mathcal{H}$*. Due to noise in LLM responses (e.g., a single mistake where the model uses world knowledge yields* $\varepsilon = 1$*), we use the 99th-percentile undercompensation rather than the maximum:* $\hat{\varepsilon}_{\mathcal{H}}(\mathbf{u}) = Q_{0.99}\big(\{\hat{v}(S) - \sum_{i \in S} u_i \ : \ S \in \mathcal{H}\}\big)$.

Lower values of $\hat{\varepsilon}_{\mathcal{H}}(\mathbf{u})$ indicate a more stable allocation, as fewer coalitions are substantially undercompensated. We additionally report Kendall's τ_b rank correlation with document-type labels (Gold $>$ Evidence $>$ Negative), gold false-negative and negative false-positive rates, % nonzero utility entries (measuring egalitarianness), and the total number of LLM calls before convergence—the primary axis on which our methods improve over prior work.

Algorithm configurations. We test three coalition seedings for the initial constraint set: random coalitions (**R**), singleton coalitions (**S**), and LLM-proposed coalitions (**L**), where the LLM proposes the minimal winning coalitions it believes are sufficient to answer the question. For the separation oracle, we use batches of 2^6 coalitions per round: the random oracle (**R**) samples uniformly, while the LLM oracle (**L**) proposes coalitions it suspects are under-compensated. As baselines, we evaluate **YP** (random sampling + LP solve), **YP S** (with singletons), and **ZS** (single-call zero-shot).

We set a maximum budget of $B = 2^{12}$ checked coalitions per method. **YP** exhausts its full budget, then solves the LP; **CG** methods terminate early when no violated constraint is found. All

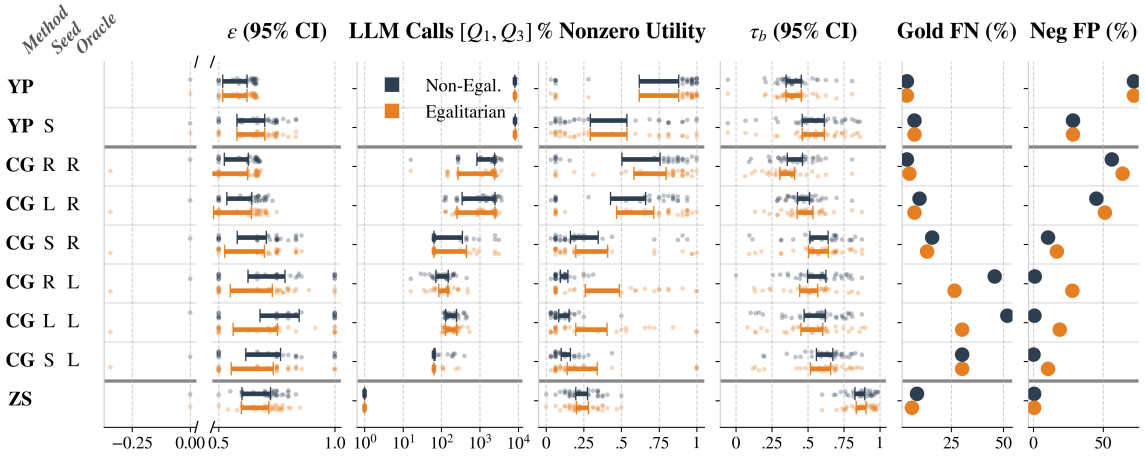


Figure 1: Credit assignment results on BrowseComp-Plus (n=32) for non-egalitarian (dark) and egalitarian (light) settings. Bars show 95% bootstrap CIs of the sample means; points are individual problem instances. See Figure 3 for the analogous Synthetic results.

experiments use batches of 64 coalitions per **CG** round. We repeat across 32 problem instances per dataset. Aggregate results are summarized in Figure 1.

Aggregate findings. Across both datasets, **CG** methods with a random oracle achieve comparable or better holdout ε than **YP** while using 3–17× fewer calls. On BrowseComp-Plus, **CG(R,R)** achieves $\hat{\varepsilon}_{\mathcal{H}} \approx 0.64$ (matching **YP**) at median ∼2200 calls vs. 8192. On Synthetic, **CG(R,R)** achieves $\hat{\varepsilon}_{\mathcal{H}} \approx 0.47$ vs. 0.48 for **YP**, with similar call savings. LLM-oracle **CG** variants converge fastest (median 64–145 calls) but at moderately higher ε (0.51–0.80). Singleton seeding (**S**) is effective for rapid convergence, establishing each document's standalone value in the first round.

An important pattern emerges in the subsidiary metrics. LLM-oracle **CG** methods produce sparser allocations (% nonzero ≈ 12–25%), concentrating utility on fewer documents. This yields low negative FP rates (≤2%) but higher gold FN rates (up to 47%). The sparsity arises because the LLM oracle targets coalitions it expects to be high-value, so the LP only sees constraints involving genuinely useful documents and concentrates credit accordingly. By contrast, **YP**'s random sampling includes coalitions where negative documents appear alongside gold or evidence documents. When such coalitions happen to produce correct answers, credit leaks to the negatives. This explains **YP**'s broad-but-imprecise allocations (% nonzero ≈ 77%, negative FP 72%, gold FN 3.6%). The egalitarian QP objective successfully redistributes credit, roughly doubling % nonzero for **CG** methods and reducing gold FN rates at the cost of increased negative FP, without degrading ε.

Individual convergence traces. To understand the optimization dynamics, we present representative traces in Figure 2. We use **YP**'s final holdout $\hat{\varepsilon}_{\mathcal{H}}$ as a reference point for each instance, since **YP** exhausts its full budget and provides the strongest baseline. In the Synthetic n=32 instance (Figure 2, left), **YP** converges to $\hat{\varepsilon}_{\mathcal{H}} = 0.65$ after all 8,192 calls. **CG(R,R)** matches this baseline, reaching $\hat{\varepsilon}_{\mathcal{H}} \approx 0.65$ (within 0.01) in only ∼1,900 calls, a 4× reduction. The LLM-oracle variants **CG(L,R)** and **CG(L,S)** converge even faster at 145 and 119 calls respectively, though they overshoot the baseline by ∼0.10 ($\hat{\varepsilon}_{\mathcal{H}} \approx 0.75$). **ZS** overshoots further ($\hat{\varepsilon}_{\mathcal{H}} \approx 0.80$, +0.15 above **YP**). The reported ε (dashed) is biased downward since it reflects only constraints seen so far, so we track both.

References

[1] Torin Anderson and Shuo Niu. Making AI-enhanced videos: Analyzing generative AI use cases in YouTube content creation. In *Proceedings of the Extended Abstracts of the CHI Conference on Human Factors in Computing Systems*, pages 1–7, 2025.

[2] Maria-Florina Balcan, Ariel D Procaccia, and Yair Zick. Learning cooperative games. In *Proceedings of the 24th International Joint Conference on Artificial Intelligence*, pages 475–481, 2015.

[3] Eric Balkanski, Umar Syed, and Sergei Vassilvitskii. Statistical cost sharing. *Advances in Neural Information Processing Systems*, 30, 2017.

[4] Yurui Chang, Bochuan Cao, Yujia Wang, Jinghui Chen, and Lu Lin. Jopa: Explaining large language model's generation via joint prompt attribution. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 22106–22122, 2025.

[5] Zijian Chen, Xueguang Ma, Shengyao Zhuang, Ping Nie, Kai Zou, Andrew Liu, Joshua Green, Kshama Patel, Ruoxi Meng, Mingyi Su, et al. BrowseComp-Plus: A more fair and transparent evaluation benchmark of deep-research agent. *arXiv preprint arXiv:2508.06600*, 2025.

[6] Benjamin Cohen-Wang, Harshay Shah, Kristian Georgiev, and Aleksander Madry. Contextcite: Attributing model generation to context. *Advances in Neural Information Processing Systems*, 37:95764–95807, 2024.

[7] Robert W Day and Peter Cramton. Quadratic core-selecting payment rules for combinatorial auctions. *Operations Research*, 60(3):588–603, 2012.

[8] Robert W Day and Subramanian Raghavan. Fair payments for efficient allocations in public sector combinatorial auctions. *Management science*, 53(9):1389–1406, 2007.

[9] Luca De Grandisa, Francesco Maria Granataa, Davide Costaa, Antonio Lanzaa, and Ermelinda Oroa. Improving context-attribution with semi-supervised cross-encoders. *European Conference on Artificial Intelligence (ECAI)*, 2025.

[10] Guy Desaulniers, Jacques Desrosiers, and Marius M Solomon. *Column Generation*, volume 5. Springer Science & Business Media, 2006.

[11] Ian Gemp, Marc Lanctot, Luke Marris, Yiran Mao, Edgar Duéñez-Guzmán, Sarah Perrin, Andras Gyorgy, Romuald Elie, Georgios Piliouras, Michael Kaisers, et al. Approximating the core via iterative coalition sampling. In *Proceedings of the 23rd International Conference on Autonomous Agents and Multiagent Systems*, pages 669–678, 2024.

[12] Amirata Ghorbani and James Zou. Data Shapley: Equitable valuation of data for machine learning. In *International Conference on Machine Learning*, pages 2242–2251. PMLR, 2019.

[13] Donald B Gillies. Solutions to general non-zero-sum games. *Contributions to the Theory of Games*, 4(40):47–85, 1959.

[14] Google DeepMind. Gemini 3 flash model card. Technical report, Google DeepMind, December 2025. URL https://storage.googleapis.com/deepmind-media/Model-Cards/Gemini-3-Flash-Model-Card.pdf. Model card.

[15] Google DeepMind. Gemini 3 pro model card. Technical report, Google DeepMind, November 2025. URL https://storage.googleapis.com/deepmind-media/Model-Cards/Gemini-3-Pro-Model-Card.pdf. Model card.

[16] Martin Grötschel, László Lovász, and Alexander Schrijver. The ellipsoid method and its consequences in combinatorial optimization. *Combinatorica*, 1(2):169–197, 1981.

[17] Martin Grötschel, László Lovász, and Alexander Schrijver. *Geometric algorithms and combinatorial optimization*. Springer, 1988.

[18] Eran Hirsch, Aviv Slobodkin, David Wan, Elias Stengel-Eskin, Mohit Bansal, and Ido Dagan. Laquer: Localized attribution queries in content-grounded generation. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15355–15370, 2025.

[19] Ruoxi Jia, David Dao, Boxin Wang, Frances Ann Hubis, Nick Hynes, Nezihe Merve Gürel, Bo Li, Ce Zhang, Dawn Song, and Costas J Spanos. Towards efficient data valuation based on the Shapley value. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pages 1167–1176. PMLR, 2019.

[20] P Kanna. How much research is being written by large language models. *Human-Centered Artificial Intelligence. Stanford University. Retrieved August*, 16, 2024.

[21] James E Kelley, Jr. The cutting-plane method for solving convex programs. *Journal of the society for Industrial and Applied Mathematics*, 8(4):703–712, 1960.

[22] Alexander Kovalenkov and Myrna Holtz Wooders. An exact bound on epsilon for nonemptiness of epsilon cores of games. *Mathematics of Operations Research*, 26(4):654–678, 2001.

[23] Ruizhe Li, Chen Chen, Yuchen Hu, Yanjun Gao, Xi Wang, and Emine Yilmaz. Attributing response to context: A jensen-shannon divergence driven mechanistic study of context attribution in retrieval-augmented generation. *arXiv preprint arXiv:2505.16415*, 2025.

[24] Fengyuan Liu, Nikhil Kandpal, and Colin Raffel. Attribot: A bag of tricks for efficiently approximating leave-one-out context attribution. *arXiv preprint arXiv:2411.15102*, 2024.

[25] Scott M Lundberg and Su-In Lee. A unified approach to interpreting model predictions. *Advances in neural information processing systems*, 30, 2017.

[26] Ikhtiyor Nematov, Tarik Kalai, Elizaveta Kuzmenko, Gabriele Fugagnoli, Dimitris Sacharidis, Katja Hose, and Tomer Sagi. Source attribution in retrieval-augmented generation. *arXiv preprint arXiv:2507.04480*, 2025.

[27] Tri-Dung Nguyen and Lyn Thomas. Finding the nucleoli of large cooperative games. *European Journal of Operational Research*, 248(3):1078–1092, 2016.

[28] Deng Pan, Keerthiram Murugesan, Nuno Moniz, and Nitesh Chawla. Context attribution with multi-armed bandit optimization. *arXiv preprint arXiv:2506.19977*, 2025.

[29] Sara Patel, Mingxun Zhou, and Giulia Fanti. Maxshapley: Towards incentive-compatible generative search with fair context attribution. *arXiv preprint arXiv:2512.05958*, 2025.

[30] Siddharth Prasad, Maria-Florina Balcan, and Tuomas Sandholm. Weakest bidder types and new core-selecting combinatorial auctions. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pages 17206–17214, 2026.

[31] Jirui Qi, Gabriele Sarti, Raquel Fernández, and Arianna Bisazza. Model internals-based answer attribution for trustworthy retrieval-augmented generation. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 6037–6053, 2024.

[32] Lloyd S Shapley. A value for n-person games. *Contributions to the Theory of Games II*, 1953.

[33] Lloyd S Shapley. On balanced sets and cores. *Naval Research Logistics Quarterly*, 14(4): 453–460, 1967.

[34] Lloyd S Shapley. Cores of convex games. *International Journal of Game Theory*, 1(1):11–26, 1971.

[35] Jiajun Shen, Tong Zhou, Yubo Chen, Delai Qiu, Shengping Liu, Kang Liu, and Jun Zhao. Transparentize the internal and external knowledge utilization in llms with trustworthy citation. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 17858–17877, 2025.

[36] TransferLab Team. pyDVL: The python library for data valuation, Mar 2025. URL https://pydvl.org/stable/.

[37] Jiachen T Wang, Prateek Mittal, Dawn Song, and Ruoxi Jia. Data Shapley in one training run. *arXiv preprint arXiv:2406.11011*, 2024.

[38] Yanting Wang, Wei Zou, Runpeng Geng, and Jinyuan Jia. {TracLLM}: A generic framework for attributing long context {LLMs}. In *34th USENIX Security Symposium (USENIX Security 25)*, pages 3845–3864, 2025.

[39] LA Wolsey. The nucleolus and kernel for simple games or special valid inequalities for 0–1 linear integer programs. *International Journal of Game Theory*, 5(4):227–238, 1976.

[40] Yingtai Xiao, Yuqing Zhu, Sirat Samyoun, Wanrong Zhang, Jiachen T Wang, and Jian Du. Tokenshapley: Token level context attribution with shapley value. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 3882–3894, 2025.

[41] Tom Yan and Ariel D Procaccia. If you like Shapley then you'll love the core. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 5751–5759, 2021.

[42] Baolei Zhang, Haoran Xin, Yuxi Chen, Zhuqing Liu, Biao Yi, Tong Li, Lihai Nie, Zheli Liu, and Minghong Fang. Who taught the lie? responsibility attribution for poisoned knowledge in retrieval-augmented generation. *arXiv preprint arXiv:2509.13772*, 2025.

Appendix A. Related work

The least core is intractable to compute in general. As a result, there is a line of work on learning and approximating it from data. In particular, Balcan et al. [2], Balkanski et al. [3], Yan and Procaccia [41] provide sample complexity bounds for learning the core given sample evaluations of the value function on coalitions drawn from some underlying distribution; Gemp et al. [11] provide iterative convex optimization algorithms for core computation that avoid solving LPs. In contrast, we harness LLMs within constraint generation to iteratively solve the least-core LP.

Wolsey [39] describes a generic constraint generation algorithm for the core (but only discusses the separation subproblem in the abstract) and Nguyen and Thomas [27] develop constraint generation algorithms for nucleolus computation (a refinement of the core that is computationally hard to even approximate [41]). In the realm of combinatorial auctions (which can be modeled as a type of coalitional game), constraint generation methods for computing core outcomes have been developed and successfully deployed in real-world high-stakes auctions [7, 8, 30].

Adaptations of the Shapley value [32] to determine the value of different features and data points in machine learning settings have become popular in recent years (e.g., [12, 19, 25, 37]). Yan and Procaccia [41] argue for the use of the core in such settings instead.

There is also a recent line of work on *context attribution*, which aims to increase interpretability by identifying the parts of the context that led to a model's generated output [4, 6, 9, 18, 23, 24, 26, 28, 29, 31, 35, 38, 40, 42]. Like these papers, we are interested in credit assignment for data which appears in-context. However unlike this line of work, we are motivated by economic considerations, not interpretability. Our credit assignment algorithms based on the least core explicitly prioritize the *incentives* of the creators, which is a critical aspect missing from prior work.

Appendix B. Sensitivity of the least core

This section bounds the difference between the solutions ε^* and $\widehat{\varepsilon}$ of LP (1) and LP (2) as a function of the true and estimated value functions. We begin with the general setting, before specializing to binary rewards. Results for supermodular value functions are included in the Appendix.

While one can always upper-bound $|\varepsilon^* - \widehat{\varepsilon}|$ with the worst-case deviation $\max_{S \subseteq N} |v(S) - \widehat{v}(S)|$, this bound may be overly pessimistic, as it depends only on the single most misestimated coalition. In contrast to this worst-case viewpoint, we use a dual characterization in terms of *balanced* distributions over coalitions. (A balanced distribution is one such that all creators have equal marginal inclusion probabilities.) This perspective allows us to obtain a refined sensitivity bound that weights estimation errors according to a worst-case balanced distribution.

Theorem 6 *Let* $\Lambda = \{\boldsymbol{\lambda} \in [0,1]^{2^N} : \sum_{S \subseteq N} \lambda_S = 1, \sum_{S \ni i} \lambda_S = \sum_{S \ni j} \lambda_S \; \forall i, j \in N\}$ *denote the set of balanced distributions over coalitions. Then,* $|\varepsilon^* - \widehat{\varepsilon}| \leq \max_{\boldsymbol{\lambda} \in \Lambda} \sum_{S \subseteq N} \lambda_S |v(S) - \widehat{v}(S)|$.

Theorem 6 can be viewed as a quantitative analog of the celebrated *Bondareva-Shapley theorem* [33] which characterizes whether or not the core is empty based on balancedness (see also Kovalenkov and Wooders [22]). One simple consequence of Theorem 6 is that a least estimated core scheme $\widehat{\boldsymbol{u}}$ is always in a $2 \max_{S \subset N} |v(S) - \widehat{v}(S)|$-expansion of the least core:

Corollary 7 $\widehat{\mathsf{core}}_{\widehat{\varepsilon}} \subseteq \mathsf{core}_{\widehat{\varepsilon}+\delta} \subseteq \mathsf{core}_{\varepsilon^*+2\delta}$, *where* $\delta = \max_{S \subseteq N} |v(S) - \widehat{v}(S)|$.

In the binary setting (where $v(S)$ and $\widehat{v}(S)$ take values in $\{0,1\}$), small pointwise disagreement between v and $\widehat{v}$ does not imply small error in the least-core deficit. In particular, even if v and $\widehat{v}$ disagree on an exponentially-small fraction of coalitions, the resulting least-core deficits ε^* and $\widehat{\varepsilon}$ may differ substantially. The following example illustrates this phenomenon.

Example 1 *Suppose $n = |N|$ is even, and let $A = \{1, \ldots, n/2\}$ and $B = \{n/2+1, \ldots, n\}$. Define $v(S) = \mathbf{1}[A \subseteq S]$ and $\widehat{v}(S) = \mathbf{1}[A \subseteq S \wedge B \subseteq S]$. Then $v(S) \neq \widehat{v}(S)$ if and only if $S = B \cup T$ for some $T \subseteq N \setminus A$. The fraction of coalitions on which v and $\widehat{v}$ disagree is therefore $\frac{2^{n/2}}{2^n-1} \approx 2^{-n/2}$. Despite this exponentially small disagreement, the least-core deficits differ significantly: $\varepsilon^* = 0$ but $\widehat{\varepsilon} = 1/2$.*

This example demonstrates that the fraction of coalitions on which v and $\widehat{v}$ disagree is not the right quantity for controlling the sensitivity of the least-core deficit in the binary setting. Instead, what matters is how these disagreements are distributed across coalitions. Instantiating the duality-based bound from Theorem 6 yields a sharper characterization in the binary setting: the sensitivity is governed by the worst-case disagreement probability under a balanced distribution over coalitions.

Corollary 8 *Let Λ be the set of balanced distributions over coalitions defined in Theorem 6. Then, $|\varepsilon^* - \widehat{\varepsilon}| \leq \max_{\boldsymbol{\lambda} \in \Lambda} \sum_{S: v(S) \neq \widehat{v}(S)} \lambda_S$.*

Appendix C. Discussion: Cooperative solution concepts

The Shapley value and the core are two of the most canonical methods for determining payments in coalitional game settings. The Shapley value is notable for being the is the only allocation that simultaneously satisfies the following four properties: efficiency (the sum of Shapley values equals the total value), symmetry (equivalent agents receive equivalent values), linearity (if the value of every coalition is a linear combination of two value functions, then each player's Shapley value is the same linear combination of their Shapley values under those functions), and null player (adding a dummy player does not change the Shapley values of others). However it is generally not an equilibrium credit allocation scheme.

In contrast, the least core may be viewed as the cooperative analog of the Nash equilibrium, as under core payments the incentive of any subset of players to unilaterally deviate is minimized. The egalitarian least core also satisfies all Shapley properties, with the exception of linearity. We choose to use the least core instead of the Shapley values as our solution concept, as the core better aligns with our goal of aligning creator incentives by preventing any subset of creators from being systematically under-compensated for their contribution to the AI-generated content item.

Appendix D. Appendix for Section B: Sensitivity of the least core

Theorem 6 *Let $\Lambda = \{\boldsymbol{\lambda} \in [0,1]^{2^N} : \sum_{S \subseteq N} \lambda_S = 1, \sum_{S \ni i} \lambda_S = \sum_{S \ni j} \lambda_S \; \forall i, j \in N\}$ denote the set of balanced distributions over coalitions. Then, $|\varepsilon^* - \widehat{\varepsilon}| \leq \max_{\boldsymbol{\lambda} \in \Lambda} \sum_{S \subseteq N} \lambda_S |v(S) - \widehat{v}(S)|$.*

Proof By LP duality, we have

$$\varepsilon^* = \max_{\boldsymbol{\lambda}, \mu} \left\{ \sum_{S \subseteq N} \lambda_S v(S) - \mu : \sum_{S \subseteq N} \lambda_S = 1, \; \sum_{S \ni i} \lambda_S = \mu \; \forall \, i \in N, \; \lambda_S \geq 0 \; \forall \, S \subseteq N \right\}$$

and

$$\widehat{\varepsilon} = \max_{\boldsymbol{\lambda},\mu} \left\{ \sum_{S\subseteq N} \lambda_S \widehat{v}(S) - \mu : \sum_{S\subseteq N} \lambda_S = 1,\ \sum_{S\ni i} \lambda_S = \mu\ \forall\, i \in N,\ \lambda_S \geq 0\ \forall\, S \subseteq N \right\}.$$

Let $\boldsymbol{\lambda}^*, \mu^*$ be a dual optimal solution for ε^*. We have

$$\varepsilon^* = \sum_{S\subseteq N} \lambda_S^* v(S) - \mu^* \quad \text{and} \quad \widehat{\varepsilon} \geq \sum_{S\subseteq N} \lambda_S^* \widehat{v}(S) - \mu^*.$$

So,

$$\varepsilon^* - \widehat{\varepsilon} \leq \sum_{S\subseteq N} \lambda_S^* (v(S) - \widehat{v}(S)).$$

Symmetrically, considering a dual optimal solution $\widehat{\boldsymbol{\lambda}}, \widehat{\mu}$ for $\widehat{\varepsilon}$, we get

$$\widehat{\varepsilon} - \varepsilon^* \leq \sum_{S\subseteq N} \widehat{\lambda}_S (\widehat{v}(S) - v(S)).$$

Therefore

$$|\varepsilon^* - \widehat{\varepsilon}| \leq \max \left\{ \sum_{S\subseteq N} \lambda_S^* (v(S) - \widehat{v}(S)), \sum_{S\subseteq N} \widehat{\lambda}_S (\widehat{v}(S) - v(S)) \right\} \leq \max_{\boldsymbol{\lambda}\in\Lambda} \sum_{S\subseteq N} \lambda_S |v(S) - \widehat{v}(S)|.$$

■

D.1. Supermodular value functions

The general sensitivity bound developed above applies to arbitrary value functions, but it does not directly yield a constructive procedure for computing approximately stable allocations. In this subsection, we show that when the value function exhibits additional structure—specifically, supermodularity—it is possible to obtain approximate core allocations in closed form, without explicitly solving the least-core optimization problem.

Supermodularity corresponds to a setting with increasing marginal returns: adding a creator to a larger coalition provides at least as much incremental value as adding them to a smaller one. Supermodularity implies that the core is non-empty in the exact case. We show that this structure can also be leveraged in the approximate setting.

Proposition 9 *Suppose v is supermodular, i.e. for all $S, T \subseteq N$, $v(S) + v(T) \leq v(S \cup T) + v(S \cap T)$. Let $S_i = \{1, \ldots, i\}$ and define the total variation distance along this chain by $\mathrm{TV}(v, \widehat{v}) = \frac{1}{2}\sum_{i=1}^{N} |v(S_i) - \widehat{v}(S_i)|$. Finally, define the allocation $\widehat{\boldsymbol{u}} \in \mathbb{R}^N$ by the marginal increments $\widehat{\boldsymbol{u}}_i = \widehat{v}(S_i) - \widehat{v}(S_{i-1})$. Then, $\widehat{\boldsymbol{u}} \in \mathsf{core}_{4\mathrm{TV}(v,\widehat{v})}$.*

Proof The proof is a direct generalization of the proof that a supermodular value function induces a nonempty core [34]. By definition, $\sum_{j\in N} \widehat{u}_j = \widehat{v}(N) = 1$. Let $i \in N$ be minimal such that $T \subseteq \{1, \ldots, i\}$. We show that, for all $T \subseteq N$, $\sum_{j\in T} \widehat{u}_j \geq v(T) - \sum_{j=1}^{i} |v(S_j) - \widehat{v}(S_j)| -$

$\sum_{j=1}^{i-1} |v(S_j) - \widehat{v}(S_j)|$ by induction on $|T|$ (which implies $\widehat{\boldsymbol{u}} \in \mathsf{core}_{4\mathrm{TV}(v,\widehat{v})}$). The base case of $T = \emptyset$ holds trivially. Consider now a general T. We have

$$\begin{aligned}
v(T) &\leq v(T \cup S_{i-1}) + v(T \cap S_{i-1}) - v(S_{i-1}) \\
&= v(S_i) + v(T \setminus \{i\}) - v(S_{i-1}) \\
&\leq \widehat{v}(S_i) - \widehat{v}(S_{i-1}) + v(T \setminus \{i\}) + |v(S_i) - \widehat{v}(S_i)| + |v(S_{i-1}) - \widehat{v}(S_{i-1})| \\
&= \widehat{u}_i + v(T \setminus \{i\}) + |v(S_i) - \widehat{v}(S_i)| + |v(S_{i-1}) - \widehat{v}(S_{i-1})| \\
&\leq \sum_{j \in T} \widehat{u}_j + \sum_{j=1}^{i} |v(S_j) - \widehat{v}(S_j)| + \sum_{j=1}^{i-1} |v(S_j) - \widehat{v}(S_j)|,
\end{aligned}$$

as desired, where the final inequality is due to the inductive hypothesis. ■

This result shows that a simple allocation constructed from successive marginal contributions under $\widehat{v}$ is guaranteed to be approximately stable with respect to the true value function v, with approximation quality controlled by how well $\widehat{v}$ matches v along the chain.

Next, we consider the complementary setting in which the estimated value function $\widehat{v}$ is supermodular.

Proposition 10 *Suppose $\widehat{v}$ is supermodular. Define δ and $\widehat{u}_i$ as in Proposition 9. Then, $\widehat{u} \in \mathsf{core}_\delta$.*

Proof Supermodularity of $\widehat{v}$ implies supermodularity of $\widehat{v}$ which implies $\widehat{\mathsf{core}}_0 \neq \emptyset$ with $\widehat{\boldsymbol{u}} \in \widehat{\mathsf{core}}_0$. By Corollary 7, $\widehat{\boldsymbol{u}} \in \mathsf{core}_\delta$. ■

In this case, supermodularity of $\widehat{v}$ guarantees that the constructed allocation lies exactly in the estimated core, and therefore, by the general sensitivity bound, it is also approximately stable with respect to the true value function. Taken together, these results show that supermodularity enables efficient and direct construction of approximately optimal allocations without solving the full least-core LP.

Appendix E. Appendix for Section 3: Computing the least core

E.1. Polynomial-time core computation via the ellipsoid method

We first consider the separation oracle that randomly samples m coalitions and returns an under-compensated coalition if one exists (formally defined in Algorithm 2). What follows is a guarantee on the δ-*probable* least estimated core: Theorem 11 says that with high probability, a $1 - \delta$ fraction of creator coalitions will have a value deficit that is at most as large as their deficit under the least estimated core, when the number of samples in the sampling-based oracle is large enough.

Theorem 11 *Let $\mathcal{D}$ be a distribution over coalitions. Suppose the ellipsoid method (Alg. 4) is instantiated such that* GetUnhappyCoalition *is the sampling-based separation oracle (Alg. 2) with $m = O(\log(n/\gamma)/\delta)$ samples, and let $\boldsymbol{u}^{(\mathrm{alg})}$ be the returned value-sharing scheme. Then with probability at least $1 - \gamma$, $\mathbb{P}_{S \sim \mathcal{D}}\left[\sum_{i \in S} u_i^{(\mathrm{alg})} + \widehat{\varepsilon} \geq \widehat{v}(S)\right] \geq 1 - \delta$, where $\widehat{\varepsilon}$ is the least estimated core deficit.*

It is worth comparing our results in Theorem 11 to those of Theorem 1 in Yan and Procaccia [41], which obtains a value-sharing scheme in the δ-probable least core by randomly sampling $O((n+\log(1/\gamma))/\delta^2)$ coalitions and solving the least core LP with just those coalitions' constraints. Ignoring logarithmic factors, our total sample complexity across all rounds scales as $\widetilde{O}(n^2/\delta)$ (since the ellipsoid method runs for $\widetilde{O}(n^2)$ rounds), whereas theirs scales as $\widetilde{O}(n/\delta^2)$. Hence the two bounds are complementary and are better in different regimes.

We record an important consequence of Theorem 11 for the egalitarian least estimated core. The quadratic objective in the egalitarian least core admits an exact separation oracle through its epigraph, and so the only source of randomness is the sampling-based oracle. Thus we can recover a probabilistic approximation of the egalitarian least estimated core by running two instances of the ellipsoid method: one for the estimated least core deficit LP and one for the egalitarian least estimated core QP.

Corollary 12 *Let $\varepsilon^{(\mathrm{alg})}$ denote the estimated core deficit from Theorem 11, and consider running the ellipsoid method with the same setup as Theorem 11 to solve $\min_{\boldsymbol{u}' \in \widehat{\mathsf{core}}_{\varepsilon^{(\mathrm{alg})}}} \frac{1}{2}\|\boldsymbol{u}'\|_2^2$. Then with probability at least $1-2\gamma$, the returned solution $\boldsymbol{u}^{(\mathrm{egal})}$ minimizes $\|\boldsymbol{u}\|_2^2$ among all feasible value-sharing schemes.*

E.2. Coalition seeding for binary value functions

We now study how the choice of the initial constraint set in CG (Alg. 1, line 1) affects performance in the special case of binary rewards. Binary rewards are important in practice (e.g. "Does a set of documents suffice to answer a query?"), and admit additional structure that can be exploited algorithmically.

Definition 13 (Minimal winning coalition) *Let $\widehat{v} : 2^N \to \{0,1\}$ be a (estimated) binary value function that is* monotone*, i.e. $\widehat{v}(T) \geq \widehat{v}(S)$ for all $T \supseteq S$. A* minimal winning coalition (MWC) *for $\widehat{v}$ is a coalition $S \subseteq N$ such that $\widehat{v}(S) = 1$, but $\widehat{v}(T) = 0$ for all strict subsets $T \subset S$.*

Intuitively, MWCs are the smallest sets of creators that can independently generate value. A key structural property is that MWCs form a complete description of the core constraints: any winning coalition contains at least one MWC, and constraints for larger coalitions are redundant once MWCs are enforced.

Lemma 14 *Given as input a list of all MWCs $\{S_1, \ldots, S_k\}$ of $\widehat{v}$, a least estimated core allocation can be found in $\mathsf{poly}(n,k)$ time by solving LP (3). Furthermore, the egalitarian least core can be computed in $\mathsf{poly}(n,k)$ time by solving QP (4).*

In practice, however, we rarely know all MWCs in advance. The next result shows that CG can efficiently recover any missing MWCs, provided the separation oracle identifies highly violated coalitions.

Theorem 15 *Let $\mathcal{L} = \{S_1, \ldots, S_k\}$ denote the list of MWCs that CG is seeded with and let $\eta = |\{T \subset N : T \text{ is a MWC}, T \notin \mathcal{L}\}|$ be the number of missing MWCs. Suppose we are given access to a* max-violation separation oracle*, i.e. $\mathrm{GetUnhappyCoalition}(\boldsymbol{u}) \in \mathrm{argmin}_{S:\widehat{v}(S)=1} \sum_{i \in S} u_i$. Then Algorithm 1 computes the egalitarian least estimated core in $\mathsf{poly}(n, k+\eta)$ runtime with $\eta+1$ oracle calls.*

Input: number of samples m, value function estimate $\widehat{v}$, value-sharing scheme $\boldsymbol{u}$, $\varepsilon \geq 0$
for $j = 1, \ldots, m$ **do**
  Sample $S_j \sim \mathcal{D}$
  Return S_j if $\sum_{i \in S_j} u_i + \varepsilon < \widehat{v}(S_j)$
end

Algorithm 2: GetUnhappyCoalition: sampling-based

Input: number of samples m, value function estimate $\widehat{v}$, value-sharing scheme u, $\varepsilon \geq 0$
for $j = 1, \ldots, m$ **do**
  Ask the LLM which subset it believes is most violated given u. Denote this subset by S_j.
  Return S_j if $\sum_{i \in S_j} u_i + \varepsilon < \widehat{v}(S_j)$
end

Algorithm 3: GetUnhappyCoalition: LLM-based

These results suggest that effective seeding strategies for binary rewards should aim to approximate the set of MWCs, as the number of CG iterations is governed by how many MWCs are missing. Motivated by this, we experiment with several different coalition seeding strategies in Section 4.

E.3. Proofs

Theorem 11 *Let $\mathcal{D}$ be a distribution over coalitions. Suppose the ellipsoid method (Alg. 4) is instantiated such that* GetUnhappyCoalition *is the sampling-based separation oracle (Alg. 2) with $m = O(\log(n/\gamma)/\delta)$ samples, and let $\boldsymbol{u}^{(\text{alg})}$ be the returned value-sharing scheme. Then with probability at least $1-\gamma$, $\mathbb{P}_{S\sim\mathcal{D}}\left[\sum_{i\in S} u_i^{(\text{alg})} + \widehat{\varepsilon} \geq \widehat{v}(S)\right] \geq 1-\delta$, where $\widehat{\varepsilon}$ is the least estimated core deficit.*

Proof The classical central-cut ellipsoid method in dimension d makes at most $2d(d+1)\log(R/r)$ oracle calls; in our LP, $d = n+1$, so this is at most $2(n+1)(n+2)\ln(R/r)$. Here, R is the radius of a ball containing $\mathcal{K} := \{(\boldsymbol{u}, \varepsilon) \ : \ \boldsymbol{u} \in \Delta^N, \ \varepsilon \in (0,1)\}$ and r is the radius of a ball contained within the relative interior of $\mathcal{K}$. By standard bit complexity arguments, $R/r = O(\text{poly}(n))$.

In any given round, the probability of the sampling-based separation oracle failing is at most $(1-\delta)^m$. Taking a union bound over the $O(n^2\log(R/r))$ calls to the separation oracle, the probability of failure γ is at most $O((1-\delta)^m n^2 \log(n))$. Rearranging terms and solving for m yields the desired sample size in the sampling-based separation oracle.

If the ellipsoid method terminates at $(\boldsymbol{u}^{(\text{alg})}, \varepsilon^{(\text{alg})})$ under the clean event, then it must be the case that

$$p(\boldsymbol{u}^{(\text{alg})}, \varepsilon^{(\text{alg})}) = \mathbb{P}_{S\sim D}\left[\sum_{i\in S} u_i^{(\text{alg})} + \varepsilon^{(\text{alg})} < \widehat{v}(S)\right] < \delta,$$

because otherwise the final oracle call would have returned a violated coalition instead of allowing termination. Since $\varepsilon \leq \widehat{\varepsilon}$, $\mathbb{P}_{S\sim\mathcal{D}}\left[\sum_{i\in S} u_i^{(\text{alg})} + \varepsilon^{(\text{alg})} \geq \widehat{v}(S)\right] \leq \mathbb{P}_{S\sim\mathcal{D}}\left[\sum_{i\in S} u_i^{(\text{alg})} + \widehat{\varepsilon} \geq \widehat{v}(S)\right]$, and so

$$\mathbb{P}_{S\sim\mathcal{D}}\left[\sum_{i\in S} u_i^{(\text{alg})} + \widehat{\varepsilon} \geq \widehat{v}(S)\right] \geq 1-\delta.$$

■

Algorithm 4: Ellipsoid method for the least estimated core

Set $T = 2(n+1)(n+2)\ln(R/r)$ and initialize ellipsoid E to be the ball $\{x \in \mathbb{R}^{n+1} : \|x - c\|_2 \leq R\}$, where $c = (\frac{1}{2}, \ldots, \frac{1}{2}) \in \mathbb{R}^{n+1}$ and $R = \frac{\sqrt{n+1}}{2}$.

for $t = 1, 2, \ldots, T$ **do**
- Let $(\boldsymbol{u}^{(t)}, \varepsilon^{(t)})$ be the center of E
- **if** $\sum_{i \in N} u_i^{(t)} \neq 1$ **then**
  - Use the violated hyperplane $\sum_{i \in N} u_i = 1$ to update E
  - **continue**
- **end**
- **if** $u_i^{(t)} \notin [0,1]$ *for some i, or* $\varepsilon^{(t)} \notin [0,1]$ **then**
  - Use the corresponding constraint to update E
  - **continue**
- **end**
- $S^{(t)} \leftarrow \mathrm{GetUnhappyCoalition}(\boldsymbol{u}^{(t)}, \varepsilon^{(t)})$
- **if** $S^{(t)} \neq$ `None` **then**
  - Use the violated hyperplane $\sum_{i \in S^{(t)}} u_i + \varepsilon = \widehat{v}(S^{(t)})$ to update E
  - **continue**
- **end**
- **return** $(u^{(t)}, \varepsilon^{(t)})$

end

Corollary 16 *Let $\varepsilon^{(\mathrm{alg})}$ denote the estimated core deficit from Theorem 11, and consider running the ellipsoid method with the same setup as Theorem 11 to solve $\min_{\boldsymbol{u}' \in \widehat{\mathsf{core}}_{\varepsilon^{(\mathrm{alg})}}} \frac{1}{2}\|\boldsymbol{u}'\|_2^2$. Then with probability at least $1 - 2\gamma$, the returned solution $\boldsymbol{u}^{(\mathrm{egal})}$ minimizes $\|\boldsymbol{u}\|_2^2$ among all feasible value-sharing schemes.*

Proof Introduce an auxiliary variable t and rewrite the quadratic program in epigraph form:

$$\min_{\boldsymbol{u}, t} \; t$$

subject to

$$\boldsymbol{u} \in [0,1]^N, \qquad \sum_{i \in N} u_i = 1, \qquad \sum_{i \in S} u_i \geq \hat{v}(S) - \varepsilon^{(\mathrm{alg})} \;\; \forall S \subset N, \qquad t \geq \frac{1}{2}\|\boldsymbol{u}\|_2^2.$$

This is a convex optimization problem in dimension $n+1$.

We first describe a separation oracle for the epigraph constraint $t \geq \frac{1}{2}\|\boldsymbol{u}\|_2^2$: If a candidate solution $(\boldsymbol{u}_0, t_0)$ satisfies $t_0 \geq \frac{1}{2}\|\boldsymbol{u}_0\|_2^2$, then it is feasible for the epigraph. Otherwise, since the function $f(\boldsymbol{u}) := \frac{1}{2}\|\boldsymbol{u}\|_2^2$ is convex and differentiable with gradient $\nabla f(\boldsymbol{u}_0) = \boldsymbol{u}_0$, the first-order supporting hyperplane inequality gives

$$f(\boldsymbol{u}) \geq f(\boldsymbol{u}_0) + \langle \boldsymbol{u}_0, \boldsymbol{u} - \boldsymbol{u}_0 \rangle = \boldsymbol{u}_0^\top \boldsymbol{u} - \frac{1}{2}\|\boldsymbol{u}_0\|_2^2 \qquad \forall \boldsymbol{u} \in \mathbb{R}^N.$$

Therefore every point $(\boldsymbol{u}, t)$ in the epigraph satisfies $t \geq \boldsymbol{u}_0^\top \boldsymbol{u} - \frac{1}{2}\|\boldsymbol{u}_0\|_2^2$. Since $(\boldsymbol{u}_0, t_0)$ violates $t_0 \geq \frac{1}{2}\|\boldsymbol{u}_0\|_2^2$, we have

$$\boldsymbol{u}_0^\top \boldsymbol{u}_0 - t_0 > \frac{1}{2}\|\boldsymbol{u}_0\|_2^2,$$

so $(\boldsymbol{u}_0, t_0)$ lies strictly outside the halfspace, which therefore separates it from the epigraph. Hence the quadratic epigraph admits an exact deterministic separation oracle.

The remaining constraints are the coalition constraints

$$\sum_{i \in S} u_i \geq \widehat{v}(S) - \varepsilon^{(\mathrm{alg})} \qquad \forall S \subseteq N.$$

For these, use the same sampling-based oracle as in Theorem 11, now with $\varepsilon^{(\mathrm{alg})}$ fixed. Define

$$p(\boldsymbol{u}) := \mathbb{P}_{S \sim D}\left[\sum_{i \in S} u_i + \varepsilon^{(\mathrm{alg})} < \widehat{v}(S)\right].$$

Exactly as in the proof of Theorem 4.2, if $p(\boldsymbol{u}) \geq \delta$, then one call to Algorithm 2 fails to find a violated coalition with probability at most $(1-\delta)^m$. The classical central-cut ellipsoid method makes at most $O(n^2 \log(R/r))$ oracle calls, so choosing

$$m = O\left(\frac{\log(n^2 \log n/\gamma)}{\delta}\right)$$

and taking a union bound implies that with probability at least $1-\gamma$, every oracle call behaves correctly whenever the current point violates a δ-mass of coalition constraints.

Condition on this clean event. If the ellipsoid method terminates at $(\boldsymbol{u}^{(\mathrm{egal})}, t^{(\mathrm{egal})})$, then it cannot be the case that $p(\boldsymbol{u}^{(\mathrm{egal})}) \geq \delta$, because otherwise the final oracle call would have returned a violated coalition constraint rather than allowing termination. Therefore

$$\Pr_{S \sim D}\left[\sum_{i \in S} u_i^{(\mathrm{egal})} + \varepsilon^{(\mathrm{alg})} \geq \widehat{v}(S)\right] \geq 1 - \delta.$$

This proves the δ-probable least estimated-core guarantee.

Because the optimization is carried out in epigraph form with objective $\min t$, any optimal solution satisfies

$$t^{(\mathrm{egal})} = \frac{1}{2}\|\boldsymbol{u}^{(\mathrm{egal})}\|_2^2.$$

Indeed, if strict inequality held, one could decrease t while preserving feasibility, contradicting optimality. Thus $(\boldsymbol{u}^{(\mathrm{egal})}, t^{(\mathrm{egal})})$ minimizes the ℓ_2 norm among allocations satisfying the sampled ellipsoid feasibility conditions. Taking a final union bound over both runs of ellipsoid obtains the desired result. ■

Theorem 17 *Let $\mathcal{D}$ be a distribution over coalitions. Consider running CG (Alg. 1) instantiated such that* GetUnhappyCoalition *is the sampling-based separation oracle (Alg. 2) with number of samples m, and let $\tau \leq T$ be the round on which it terminates with solution $(u^{(*,\tau)}, \varepsilon^{(*,\tau)})$. Then if $m \geq \frac{1}{\delta}\log(T/\gamma)$, with probability at least $1-\gamma$, $\mathbb{P}_{S \sim \mathcal{D}}\left[\sum_{i \in S} u_i^{(*,\tau)} + \widehat{\varepsilon} \geq \widehat{v}(S)\right] \geq 1 - \delta$. Furthermore, $\boldsymbol{u}^{(*,\tau)}$ minimizes $\|\boldsymbol{u}\|_2^2$ among all feasible value-sharing schemes defined by the $\tau - 1$ sampled constraints.*

Proof Let

$$p(u, \varepsilon) := \mathbb{P}_{S \sim \mathcal{D}} \left[\sum_{i \in S} u_i + \varepsilon < \widehat{v}(S) \right]$$

be the probability that a core constraint (under $\widehat{v}$) is violated by more than ε, and suppose we'd like to control $p(u, \varepsilon)$ such that it is at most δ. Consider the run of Algorithm 2 for round t of constraint generation and condition on the entire history up to round t. It is ok if Algorithm 2 fails if $p(u^{(*,t)}, \varepsilon^{(*,t)}) < \delta$, as this does not affect the high-probability guarantee of the δ-probable core. Therefore, assume that $p(u^{(*,t)}, \varepsilon^{(*,t)}) \geq \delta$.

The probability of Algorithm 2 failing in round t is $(1 - p(u^{(*,t)}, \varepsilon^{(*,t)}))^m \leq (1 - \delta)^m$, which we'd like to be at most γ/T. Therefore, if

$$m \geq \frac{\log(T/\gamma)}{-\log(1-\delta)} \geq \frac{\log(T/\gamma)}{\delta},$$

then one oracle call will fail with probability at most γ/T.

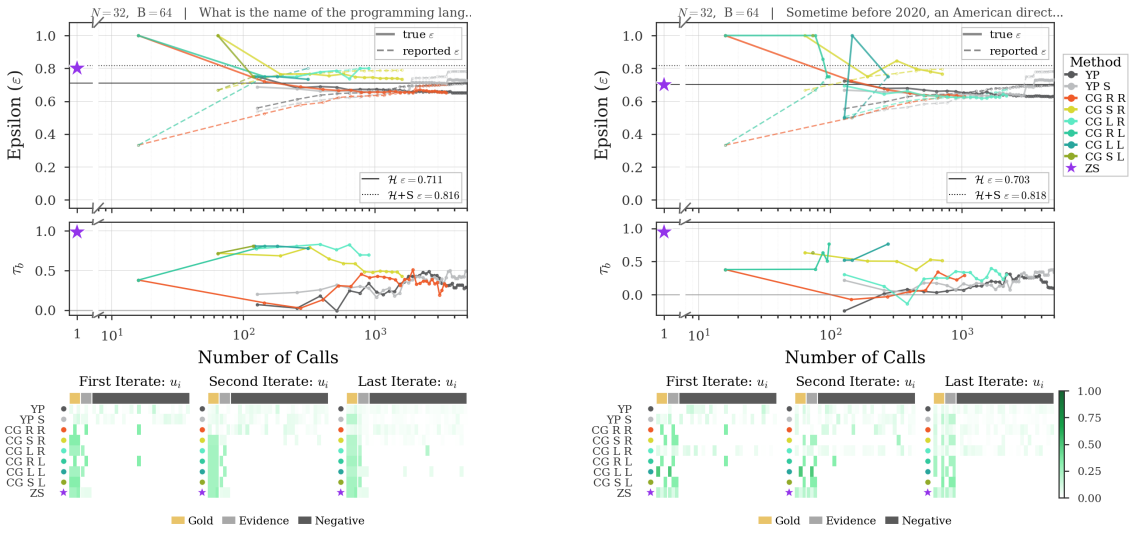
Taking a union bound over all T, we have that if $m \geq \frac{1}{\delta} \log(T/\gamma)$, then the probability of any of the T oracle calls failing is at most γ. ■

Lemma 18 *Given as input a list of all MWCs $\{S_1, \ldots, S_k\}$ of $\widehat{v}$, a least estimated core allocation can be found in* $\mathsf{poly}(n, k)$ *time by solving LP* (3)*. Furthermore, the egalitarian least core can be computed in* $\mathsf{poly}(n, k)$ *time by solving QP* (4)*.*

Proof Seed the least core LP with constraints corresponding to $S_1, \ldots, S_k$ and solve. Let S be any winning coalition, and let S_j be a MWC such that $S_j \subseteq S$. For any $\boldsymbol{u} \in [0,1]^N$, we have $\widehat{v}(S) - \sum_{i \in S} u_i = 1 - \sum_{i \in S} u_i \leq 1 - \sum_{i \in S_j} u_i = \widehat{v}(S_j) - \sum_{i \in S_j} u_i$. So, $\widehat{v}(S_j) - \sum_{i \in S_j} u_i \leq \varepsilon \Rightarrow \widehat{v}(S) - \sum_{i \in S} u_i \leq \varepsilon$ for any $\varepsilon > 0$ and therefore the constraint corresponding to S is redundant. The same reasoning extends to egalitarian least core computation (and we obtain a polynomial run-time since the quadratic program has a convex objective). ■

Theorem 15 *Let $\mathcal{L} = \{S_1, \ldots, S_k\}$ denote the list of MWCs that CG is seeded with and let $\eta = |\{T \subset N : T \text{ is a MWC}, T \notin \mathcal{L}\}|$ be the number of missing MWCs. Suppose we are given access to a* max-violation separation oracle*, i.e.* $\mathrm{GetUnhappyCoalition}(\boldsymbol{u}) \in \mathrm{argmin}_{S: \widehat{v}(S)=1} \sum_{i \in S} u_i$*. Then Algorithm 1 computes the egalitarian least estimated core in* $\mathsf{poly}(n, k+\eta)$ *runtime with $\eta + 1$ oracle calls.*

Proof Seed the least core LP with constraints corresponding to $S_1, \ldots, S_k$ and run constraint generation with the given oracle. The same reasoning as in Lemma 14 shows that for any $\boldsymbol{u}$, the oracle returns a MWC. Since there are at most η MWCs that are not already in the LP, CG terminates in at most η iterations. The same reasoning extends to egalitarian least core computation (and we obtain a polynomial run-time since the quadratic program has a convex objective). ■



(a) Synthetic, n=32

(b) BrowseComp, n=32

Figure 2: Convergence traces for representative n=32 instances. **Top panels:** holdout $\hat{\varepsilon}_{\mathcal{H}}$ (solid) and reported ε (dashed), with τ_b below. **Bottom:** credit heatmaps at the first, second, and last iterates.

Appendix F. Additional Experimental Details

Constraint generation vs. zero-shot. **ZS** is a surprisingly strong baseline: a single LLM call with no value-function feedback achieves mean $\hat{\varepsilon}_{\mathcal{H}}$ within $\sim$20% of **YP** on BrowseComp-Plus and $\sim$10% on Synthetic, with high τ_b. While this level of accuracy may suffice in some low-stakes settings, **CG** is useful when tighter ε or formal guarantees are needed: with a random separation oracle, **CG** inherits the probabilistic guarantees of Theorem 17, and matches or improves upon **YP**'s $\hat{\varepsilon}_{\mathcal{H}}$ using 3–17$\times$ fewer calls. In most settings, we recommend using LLM-seeded **CG** (**CG(L,·)**), which uses the LLM's guess as a warm start and iteratively refines it, combining the speed of **ZS** with the iterative tightening of **CG**.

Summary. Our **CG** methods achieve comparable or better holdout ε relative to **YP** while using 1–2 orders of magnitude fewer value-function queries. The egalitarian variants successfully redistribute credit more broadly with only minor impacts on ε. Additional convergence traces, n=10 experiments, and detailed dataset statistics are provided in Appendix F.

F.1. Datasets

The datasets used in our experiments comprise question–answer pairs, each associated with a set of $n \in \{10, 32\}$ documents that serve as the creators in our coalitional game.

BrowseComp-Plus [5]. This dataset assesses multi-hop question answering in a web retrieval context. Each query is paired with *gold* documents (individually sufficient to answer), *evidence* documents (each containing partial facts that yield the answer in aggregate), and *negative* documents (irrelevant or containing distracting information). When queried individually, we found that only 64% of gold documents lead to the correct answer, possibly due to our prompting strategy. However through manual inspection we observed several label quality issues: gold documents are repeated

for $t = 1, \ldots, T$ **do**
  Sample $S^t \sim \mathcal{D}$
  $\mathcal{S}^t \leftarrow \mathcal{S}^{t-1} \cup S^t$
end
Solve $\widehat{\mathrm{LP}}(\mathcal{S}^T)$; let ε be the returned optimal deficit
Solve $\widehat{\mathrm{QP}}(\mathcal{S}^T, \varepsilon)$; let $\boldsymbol{u}$ be the returned value-sharing scheme
return $\boldsymbol{u}, \varepsilon$

Algorithm 5: YP baseline

verbatim in the evidence set, some negative documents contain query-relevant facts, and some gold documents do not actually contain the answer. A detailed example is provided in Appendix F.2.

Synthetic. Given these issues with BrowseComp-Plus, we also create our own multi-hop Q/A dataset to ensure the existence of coalitions with genuine interaction effects. We prompt Gemini 3 Pro with high thinking budget to propose questions that are answerable only through a combination of 2–10 evidence documents, or a single gold document. We filter for topical uniqueness and draw negative documents from sufficiently unrelated questions. By construction, 100% of standalone gold documents lead to the correct answer.

F.2. BrowseComp-Plus Dataset Details

As an example BrowseComp-Plus query: *"A CEO who founded a company in the mid-1990s was raised in Southern Africa. His father was an engineer, and their relationship was not delightful. His first child sadly passed away as a result of SIDS. In the early 2020s, he had a child whose name had the internet buzzing. This CEO has a younger sister who co-founded a film streaming service. In an article from the early 2020s, at what age did the mother of the lady who claimed to be related to him give birth to her?"*

A *gold* document for this question explicitly states, *"...Her mother was 21..."*. A *negative* document contains information that appears relevant at first glance—e.g., *"...he shares 21-year-old twins..."*—which does not actually answer the question but contains salient keywords. An *evidence* document contains, for example, *"Tosca is the co-founder of a streaming service,"* which helps narrow down the answer.

Ideally, gold documents should receive large values in $\mathbf{u}$, evidence documents should receive nonzero but moderate values, and negative documents should receive zero. However, in our preliminary investigation, we found several violations of this intuition. In the example above, a gold document is repeated verbatim in the evidence set, violating our uniqueness assumption. Furthermore, at least one negative document contains facts relevant to the query, making it not truly negative for our purposes. We observed several similar issues: for instance, gold documents that do not contain the answer (which, in some cases, nonetheless evoke the correct response from some language models).

F.3. Synthetic n=32 Results

The aggregate Synthetic n=32 results are shown in fig. 3. Below we present selected convergence traces organized by **ZS** performance, using **YP**'s final holdout $\hat{\varepsilon}_{\mathcal{H}}$ as the reference point for each instance.

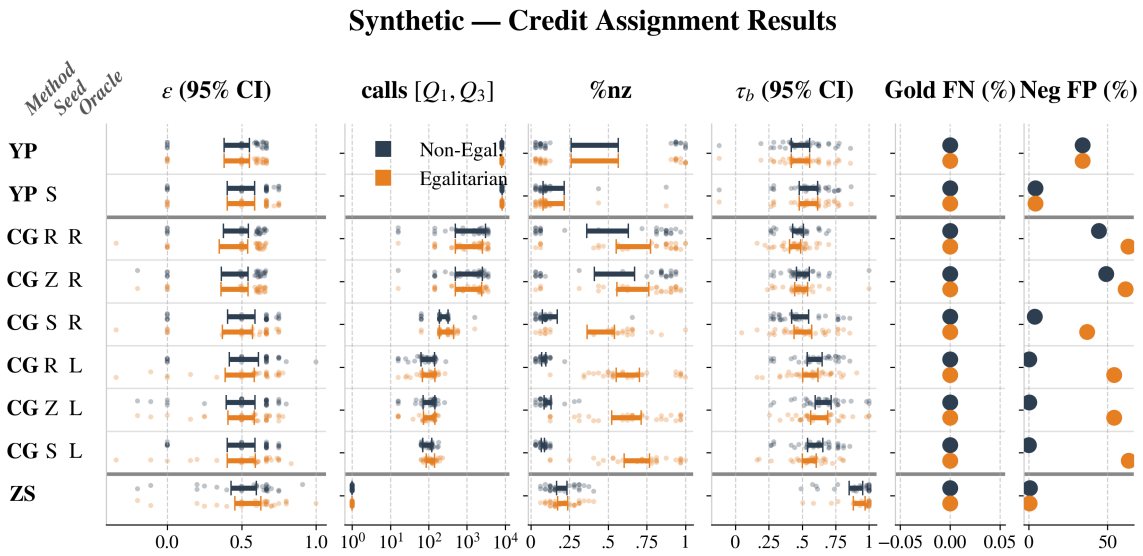
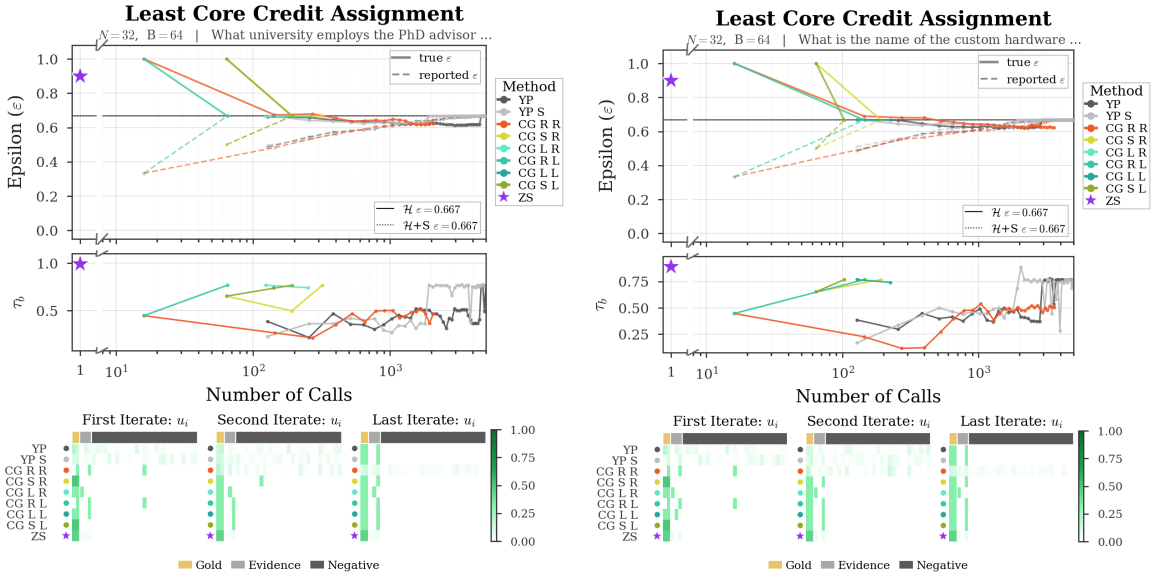


Figure 3: Credit assignment results on Synthetic (n=32) for non-egalitarian (dark) and egalitarian (light) settings. Lower $\hat{\varepsilon}_{\mathcal{H}}$ is better. Bars show 95% bootstrap CIs; points are individual problem instances. Compare with fig. 1 for BrowseComp-Plus.

F.3.1. CASES WHERE ZS FAILS

These instances (fig. 4) demonstrate that **ZS** can produce poor allocations, motivating the need for **CG**'s iterative refinement with guarantees. In Row 2, **YP** reaches $\hat{\varepsilon}_{\mathcal{H}} = 0.67$ and **CG(R,R)** matches it (within 0.04) in $\sim$2,000 calls, while **ZS** overshoots substantially at $\hat{\varepsilon}_{\mathcal{H}} = 0.90$ (+0.23 above **YP**). Row 21 shows a similar pattern: **CG(R,R)** reaches within 0.04 of **YP**'s baseline, but **ZS** overshoots by +0.23.



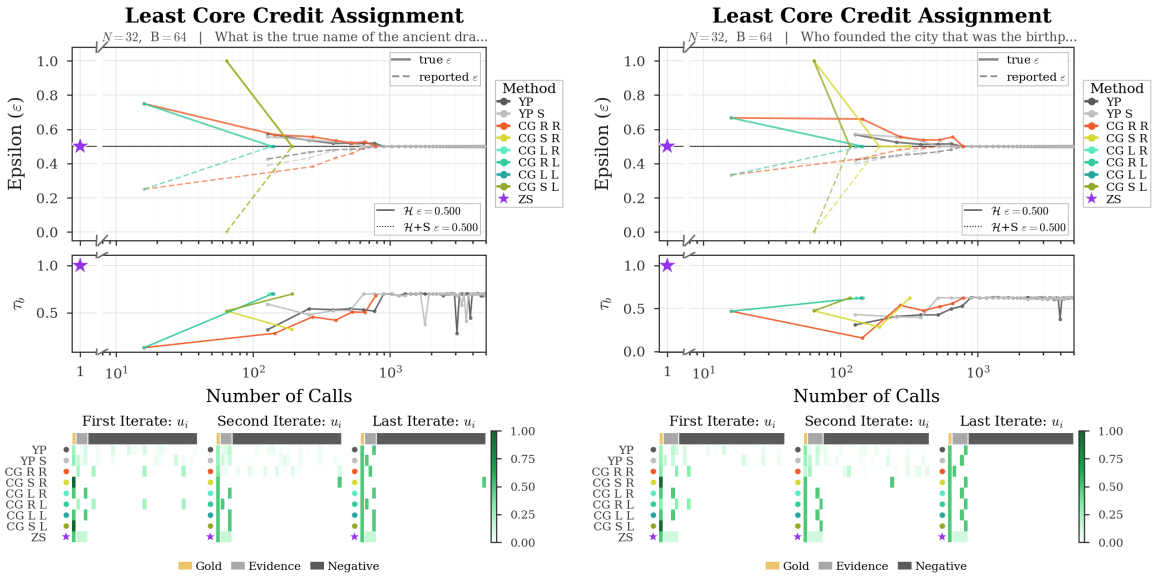
(a) Synthetic n=32, Row 2 (non-egal.)

(b) Synthetic n=32, Row 21 (non-egal.)

Figure 4: **ZS** failure cases: **ZS** overshoots **YP**'s holdout $\hat{\varepsilon}_{\mathcal{H}}$ by +0.23, while **CG(R,R)** matches the baseline with 3–4$\times$ fewer calls.

F.3.2. CASES WHERE ZS SUCCEEDS

These instances (fig. 5) show that **ZS** can sometimes produce remarkably good allocations in a single call. In Row 0, all methods—including **ZS**—match **YP**'s $\hat{\varepsilon}_{\mathcal{H}} = 0.50$ exactly, suggesting this is an easy instance. In Row 3, **ZS** again matches **YP** exactly at $\hat{\varepsilon}_{\mathcal{H}} = 0.50$, and all **CG** variants converge to the same optimum, confirming the allocation is tight.



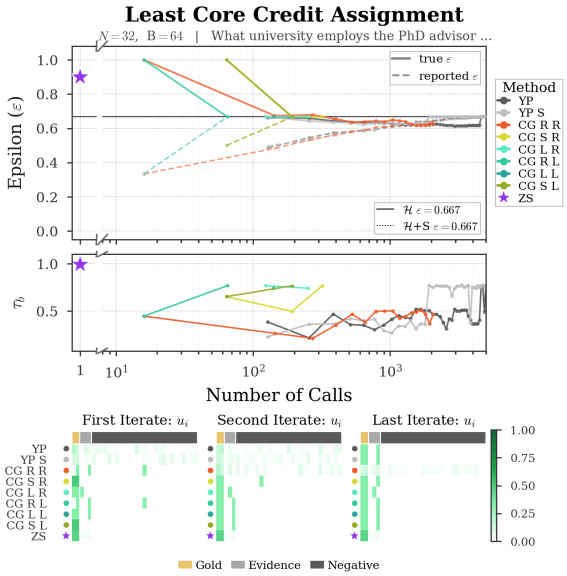
(a) Synthetic n=32, Row 0 (non-egal.)

(b) Synthetic n=32, Row 3 (non-egal.)

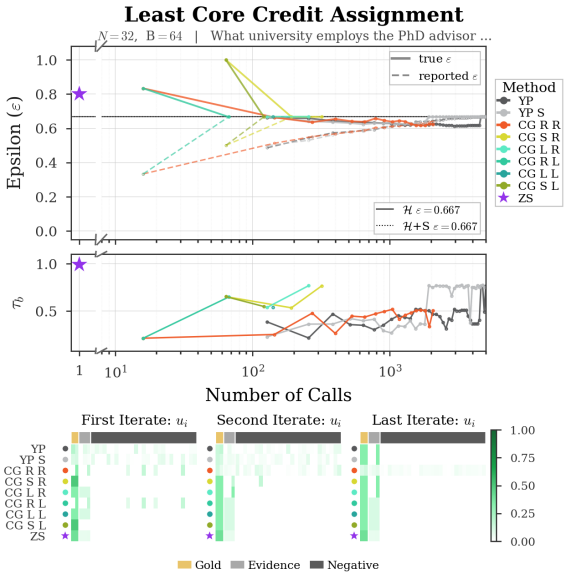
Figure 5: **ZS** success cases: **ZS** matches **YP**'s holdout $\hat{\varepsilon}_{\mathcal{H}} = 0.50$ in a single call, with high τ_b.

F.3.3. EGALITARIAN EFFECTS

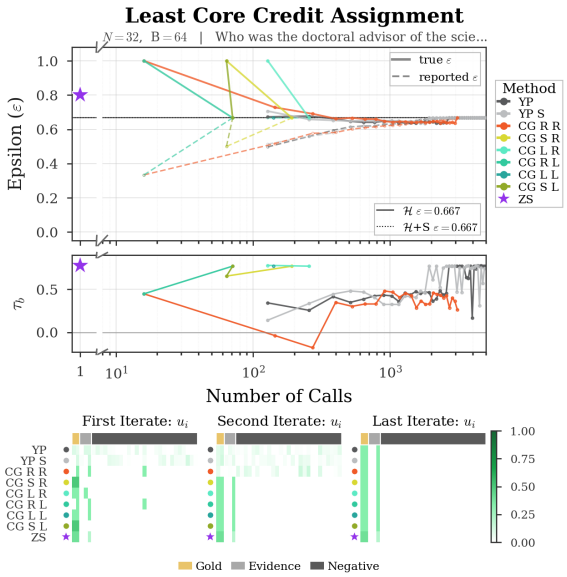
These traces (figs. 6 and 7) compare non-egalitarian (left) and egalitarian (right) variants on the same problem instance, showing how the QP objective redistributes credit without substantially degrading $\hat{\varepsilon}_{\mathcal{H}}$. In Row 2, the egalitarian **CG(R,R)** spreads utility more broadly (visible in the heatmaps) while remaining within 0.04 of **YP**'s baseline. Rows 4, 7, and 13 exhibit the same pattern: the egalitarian heatmaps show less sparse allocations, and $\hat{\varepsilon}_{\mathcal{H}}$ is either unchanged or increases by at most ~ 0.07.



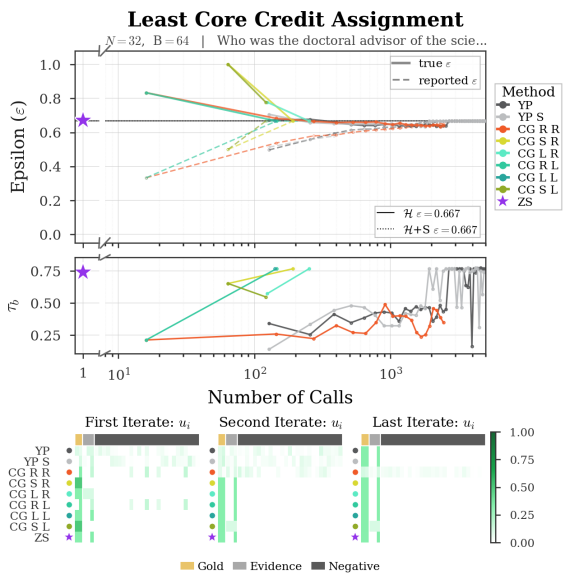
(a) Row 2 (non-egal.)



(b) Row 2 (egal.)



(c) Row 4 (non-egal.)

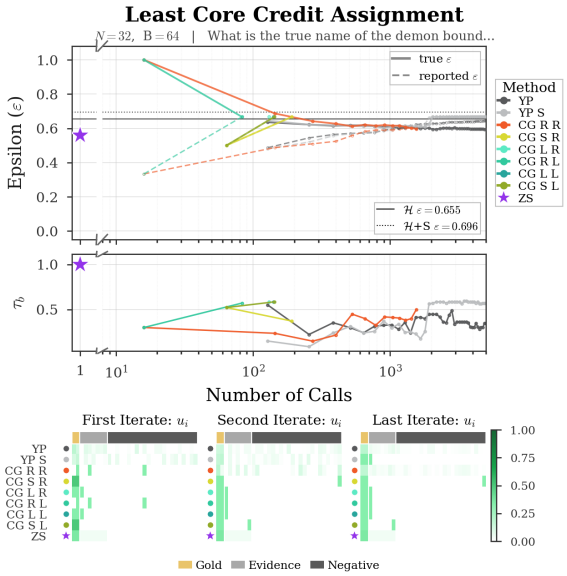


(d) Row 4 (egal.)

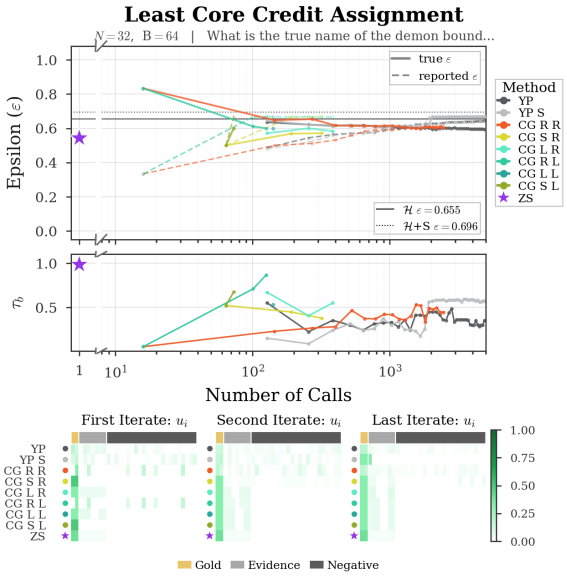
Figure 6: Egalitarian effects on Synthetic n=32: non-egalitarian (left) vs. egalitarian (right). The QP objective redistributes credit more broadly, as visible in the heatmaps, with only minor impact on $\hat{\varepsilon}_{\mathcal{H}}$ relative to **YP**'s baseline.

F.4. Additional BrowseComp n=32 Traces

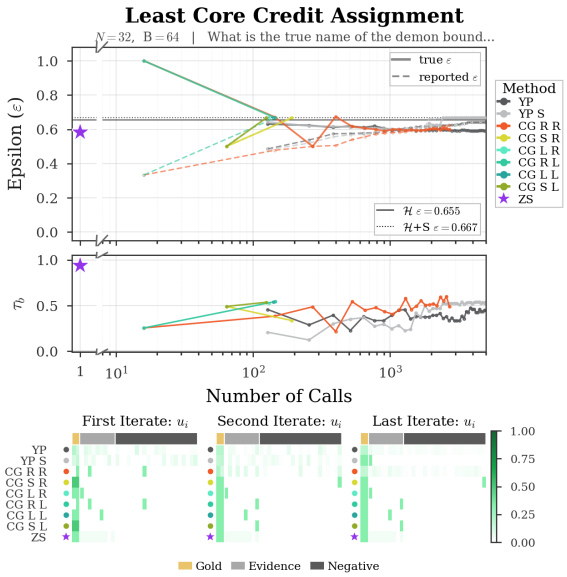
Additional BrowseComp n=32 convergence traces are shown in fig. 8, illustrating diverse **CG** dynamics across problem instances. In Row 1, **CG(R,R)** reaches within +0.01 of **YP**'s $\hat{\varepsilon}_{\mathcal{H}} = 0.65$ in ∼900 calls (a 9× reduction), while **ZS** overshoots by +0.12. Row 9 is similar: **CG(R,R)** matches



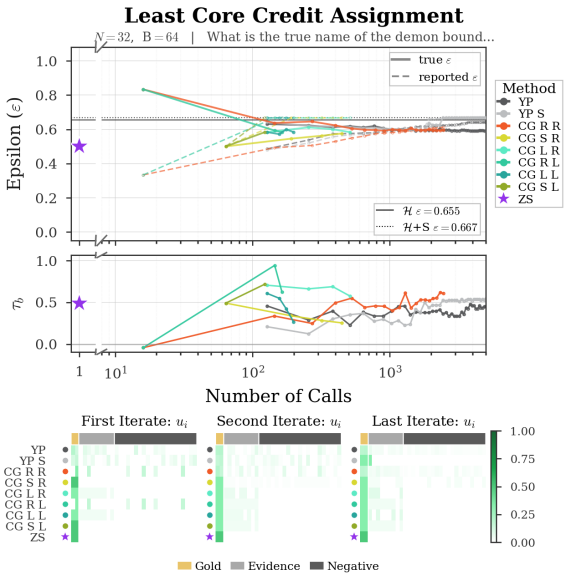
(a) Row 7 (non-egal.)



(b) Row 7 (egal.)



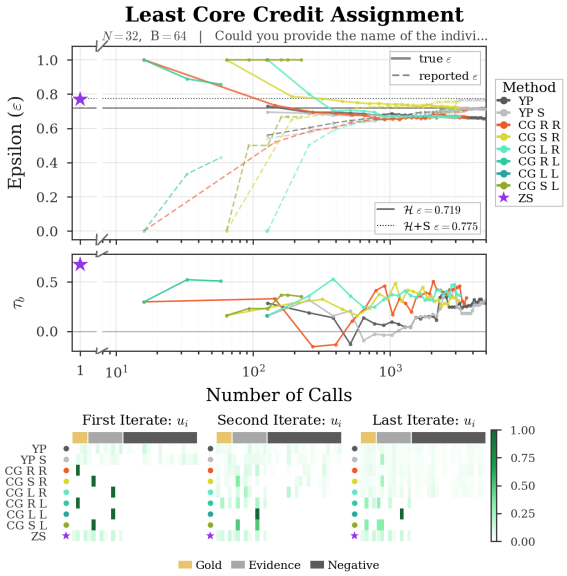
(c) Row 13 (non-egal.)



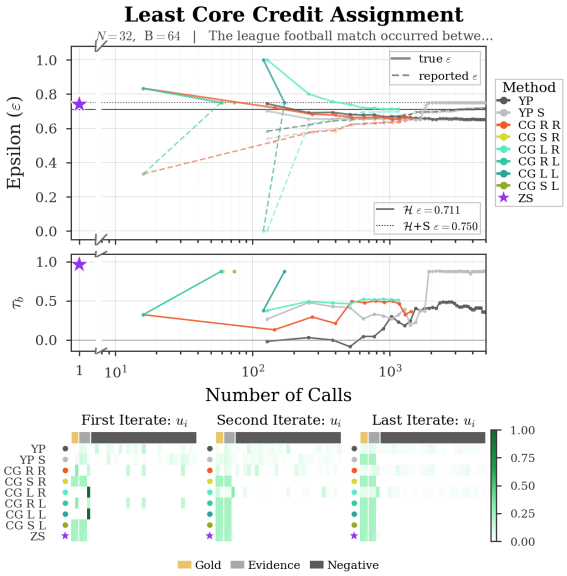
(d) Row 13 (egal.)

Figure 7: Egalitarian effects on Synthetic n=32 (continued): non-egalitarian (left) vs. egalitarian (right). Credit heatmaps are visibly less sparse under the egalitarian objective.

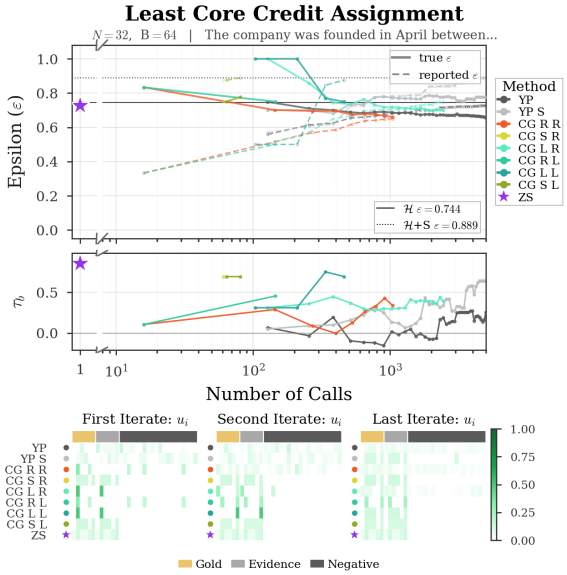
YP within +0.01 at ∼1,000 calls. Row 22 shows **CG(R,R)** matching **YP**'s $\hat{\varepsilon}_{\mathcal{H}} = 0.66$ within +0.004 in ∼1,000 calls (an 8× reduction). In Row 31, **CG(L,ZS)** and **CG(L,S)** *undershoot* **YP**'s $\hat{\varepsilon}_{\mathcal{H}} = 0.58$, achieving $\hat{\varepsilon}_{\mathcal{H}} = 0.50$ (-0.08) in only 49–64 calls.



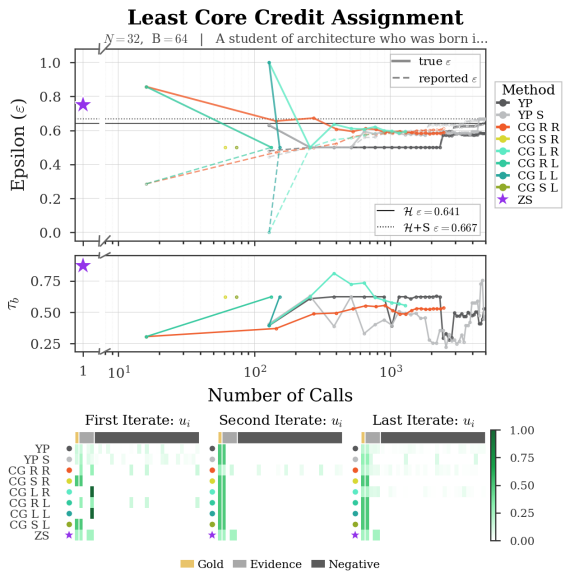
(a) BrowseComp n=32, Row 1 (non-egal.)



(b) BrowseComp n=32, Row 9 (non-egal.)



(c) BrowseComp n=32, Row 22 (non-egal.)



(d) BrowseComp n=32, Row 31 (non-egal.)

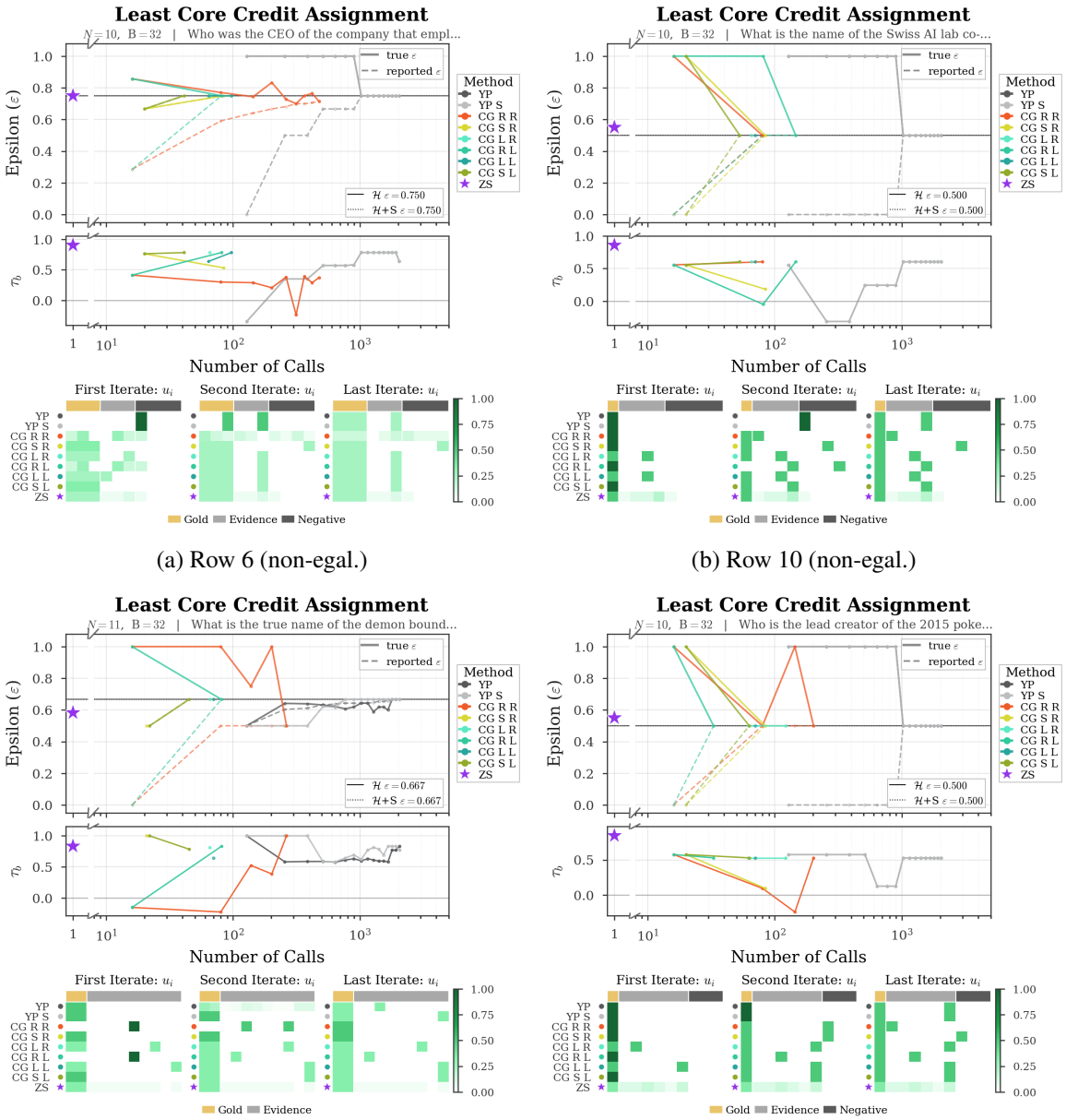
Figure 8: Additional BrowseComp n=32 convergence traces. **CG(R,R)** consistently matches **YP**'s holdout $\hat{\varepsilon}_{\mathcal{H}}$ with 8–9× fewer calls; in Row 31 the LLM-oracle variants undershoot **YP** by 0.08.

F.5. n=10 Experiments

We repeat our experiments with n=10 documents per problem instance. At this scale, the full coalition space $|2^n| = 1024$ is tractable, allowing a more precise holdout evaluation. The **YP** baseline and $\hat{\varepsilon}_{\mathcal{H}}$ are effectively exact. Forest plots and selected convergence traces for Synthetic and BrowseComp are provided in figs. 9 and 10 and figs. 11 and 12, respectively.

F.5.1. SYNTHETIC n=10

Selected non-egalitarian traces. At n=10, **CG** methods converge rapidly and often match **YP**'s holdout $\hat{\varepsilon}_{\mathcal{H}}$ exactly. In Row 10, all **CG** variants reach **YP**'s $\hat{\varepsilon}_{\mathcal{H}} = 0.50$ in 53–146 calls. In Row 13, **CG(R,R)** and **CG(R,S)** undershoot **YP**'s $\hat{\varepsilon}_{\mathcal{H}} = 0.67$ by 0.17, reaching $\hat{\varepsilon}_{\mathcal{H}} = 0.50$, while **ZS** undershoots modestly ($\hat{\varepsilon}_{\mathcal{H}} = 0.58, -0.08$). Row 6 and Row 15 show **CG(R,R)** matching or slightly beating **YP**'s baseline.



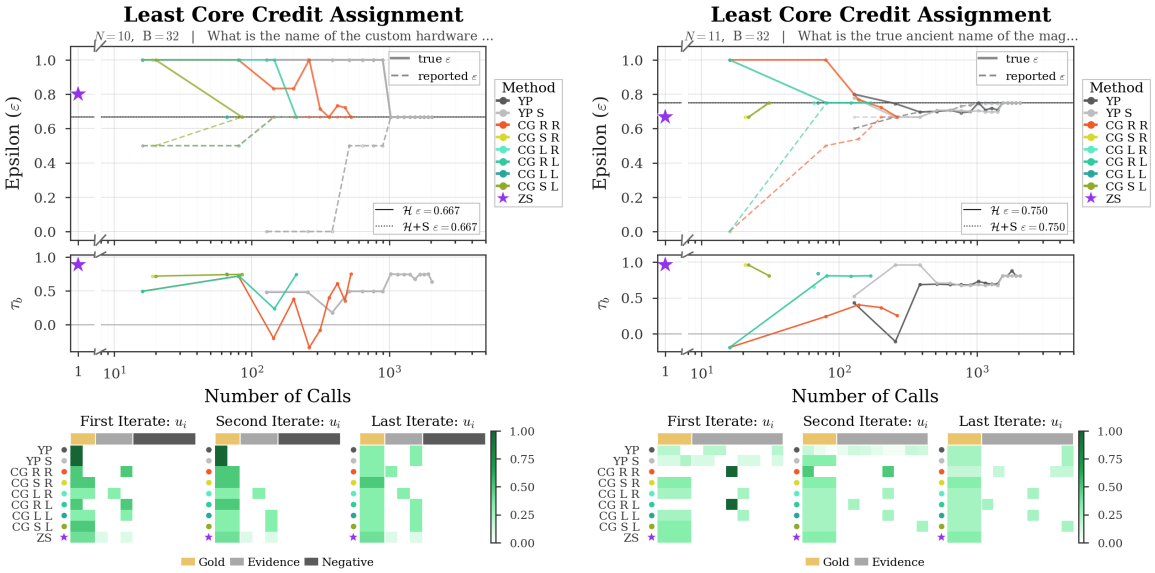
(a) Row 6 (non-egal.)

(b) Row 10 (non-egal.)

(c) Row 13 (non-egal.)

(d) Row 20 (non-egal.)

Figure 9: Synthetic n=10, non-egalitarian convergence traces. **CG** methods match or undershoot **YP**'s holdout $\hat{\varepsilon}_{\mathcal{H}}$ within ∼50–300 calls.

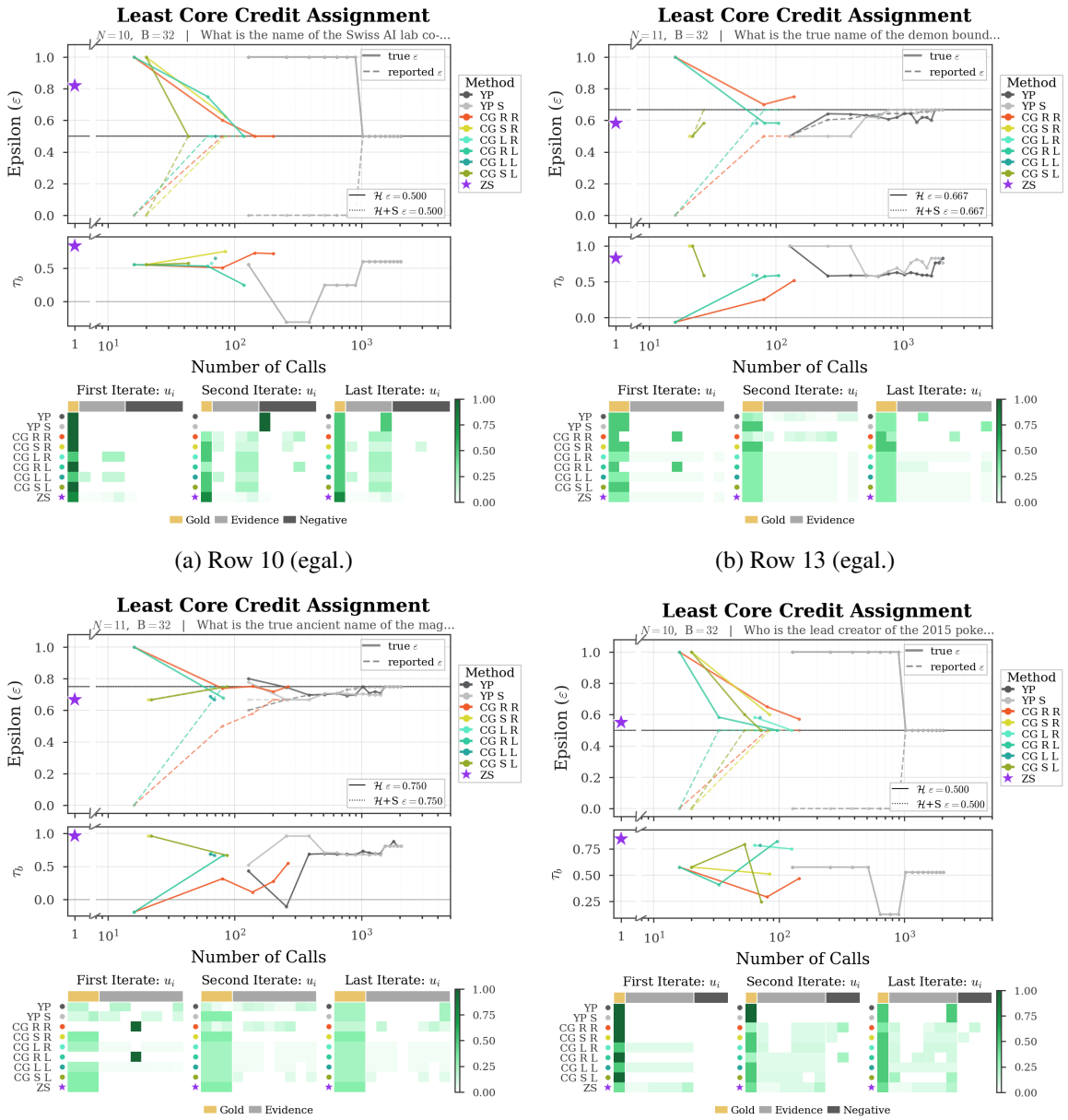


(e) Row 21 (non-egal.)

(f) Row 15 (non-egal.)

Figure 9: Synthetic n=10, non-egalitarian convergence traces (cont.).

Selected egalitarian traces. The egalitarian objective redistributes credit without substantially degrading $\hat{\varepsilon}_{\mathcal{H}}$. In Row 10 (egal.), all **CG** variants still match **YP**'s $\hat{\varepsilon}_{\mathcal{H}} = 0.50$, with visibly less sparse heatmaps. Row 15 shows **CG(R,R)** matching **YP**'s baseline within 0.01, with broader credit distribution.



(a) Row 10 (egal.)

(b) Row 13 (egal.)

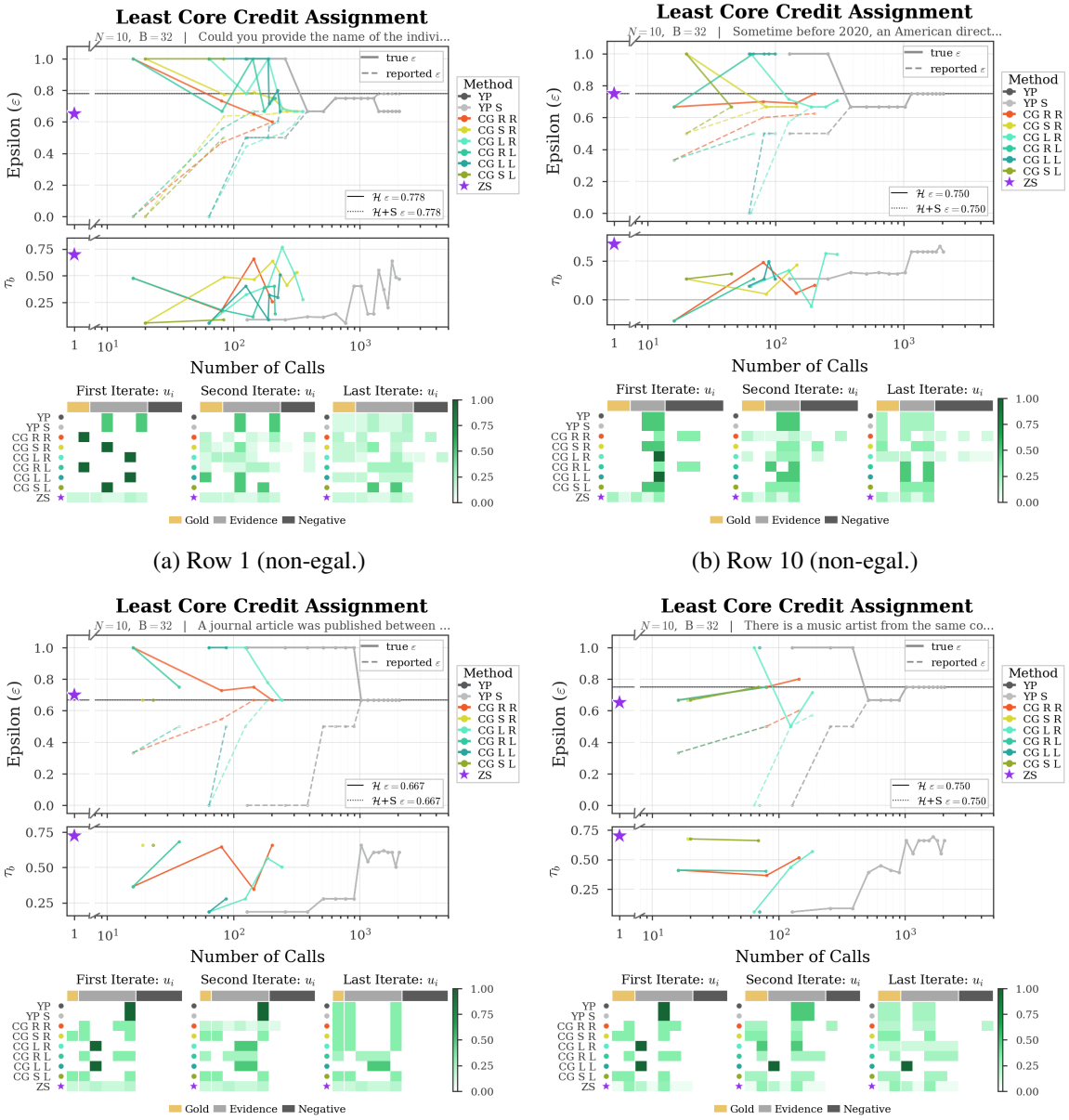
(c) Row 15 (egal.)

(d) Row 20 (egal.)

Figure 10: Synthetic n=10, egalitarian convergence traces. Heatmaps show broader credit distribution with minimal $\hat{\varepsilon}_{\mathcal{H}}$ degradation relative to **YP**.

F.5.2. BROWSECOMP n=10

Selected non-egalitarian traces. BrowseComp at n=10 shows noisier dynamics due to label-quality issues. In Row 18, **CG(R,R)** matches **YP**'s $\hat{\varepsilon}_{\mathcal{H}} = 0.67$ exactly in ∼200 calls (a 10× reduction). In Row 21, **CG(R,ZS)** undershoots **YP**'s $\hat{\varepsilon}_{\mathcal{H}} = 0.75$ by 0.05. Row 22 shows **CG(R,R)** undershooting **YP**'s $\hat{\varepsilon}_{\mathcal{H}} = 0.80$ by ∼0.09 in ∼420 calls. Row 1 demonstrates **CG(R,R)** matching **YP**'s $\hat{\varepsilon}_{\mathcal{H}} = 0.75$ exactly within ∼80 calls.



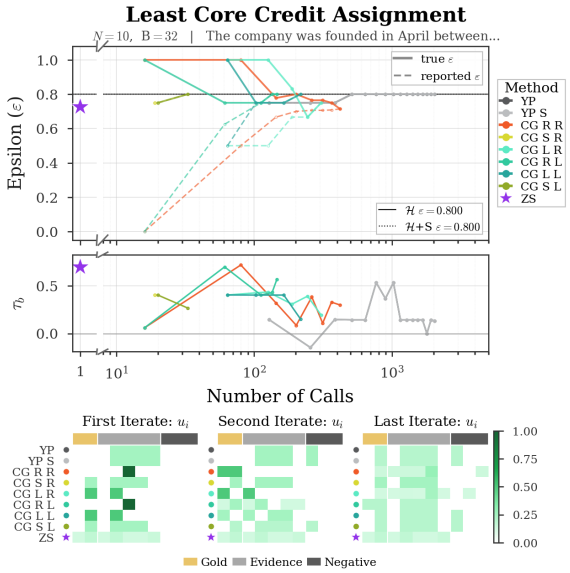
(a) Row 1 (non-egal.)

(b) Row 10 (non-egal.)

(c) Row 18 (non-egal.)

(d) Row 21 (non-egal.)

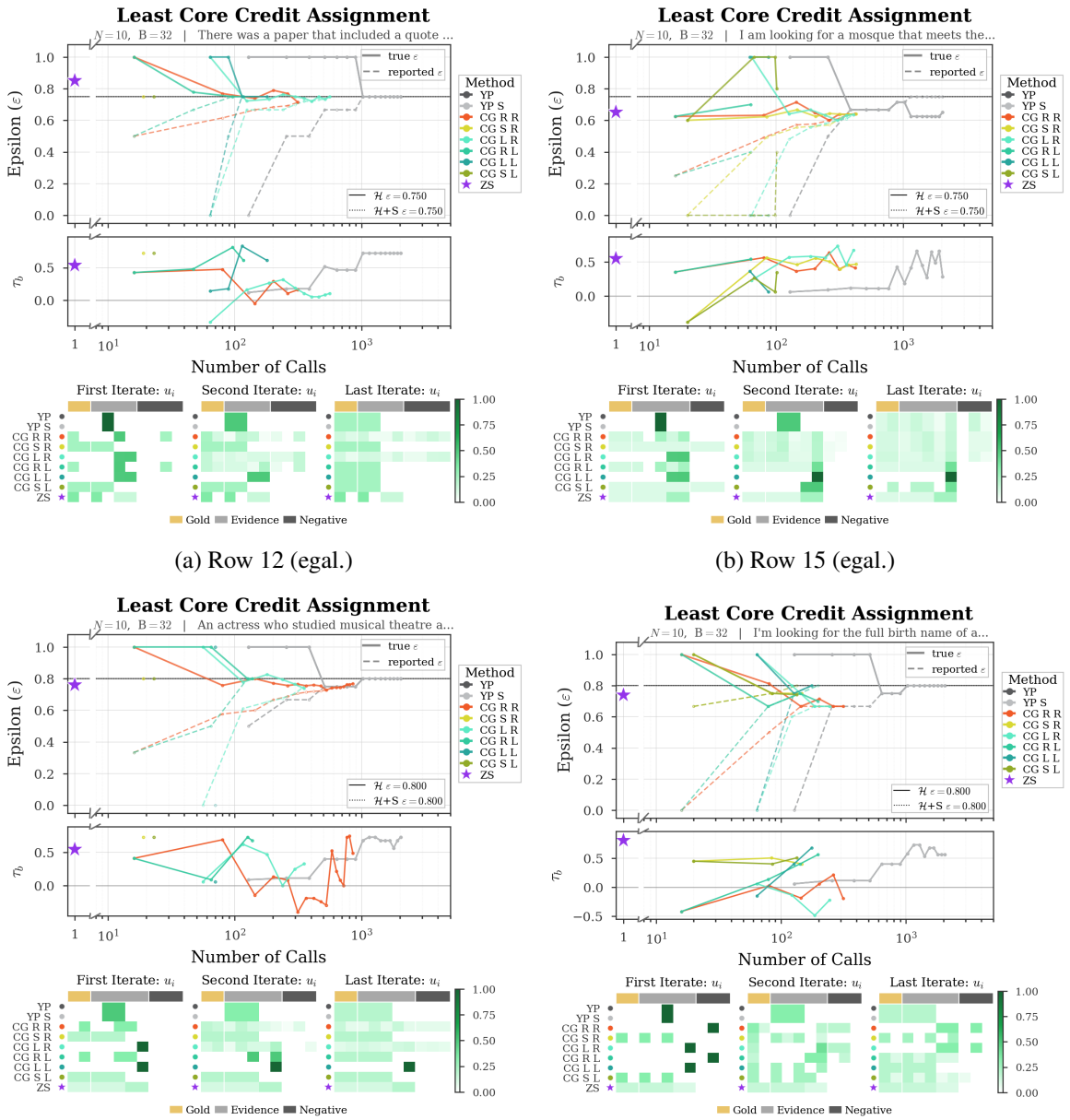
Figure 11: BrowseComp n=10, non-egalitarian convergence traces. **CG** methods match or undershoot **YP**'s holdout $\hat{\varepsilon}_{\mathcal{H}}$ in 80–420 calls.



(e) Row 22 (non-egal.)

Figure 11: BrowseComp n=10, non-egalitarian convergence traces (cont.).

Selected egalitarian traces. The egalitarian traces show broader credit redistribution with minimal $\hat{\varepsilon}_{\mathcal{H}}$ degradation. In Row 26, **CG(R,R)** undershoots **YP**'s $\hat{\varepsilon}_{\mathcal{H}} = 0.80$ by 0.13 ($\hat{\varepsilon}_{\mathcal{H}} = 0.67$ at $\sim$144 calls). Row 25 shows **CG(R,R)** slightly undershooting **YP** (-0.03). In Row 28, **CG(R,ZS)** matches **YP**'s $\hat{\varepsilon}_{\mathcal{H}} = 0.69$ within 0.01, while the LLM-oracle variants overshoot by $\sim$0.31—a reminder that **CG(L,·)** trades convergence speed for precision.



(a) Row 12 (egal.)

(b) Row 15 (egal.)

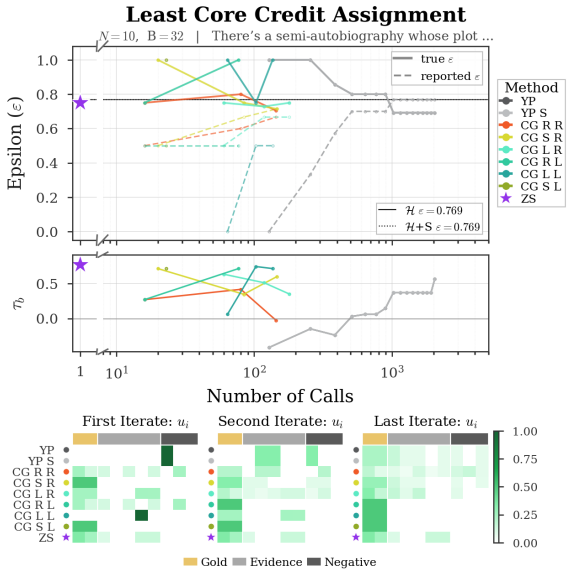
(c) Row 25 (egal.)

(d) Row 26 (egal.)

Figure 12: BrowseComp n=10, egalitarian convergence traces. Heatmaps show broader credit redistribution; **CG(R,·)** methods match or undershoot **YP**'s baseline.

F.6. Calls-to-Target CDF Plots

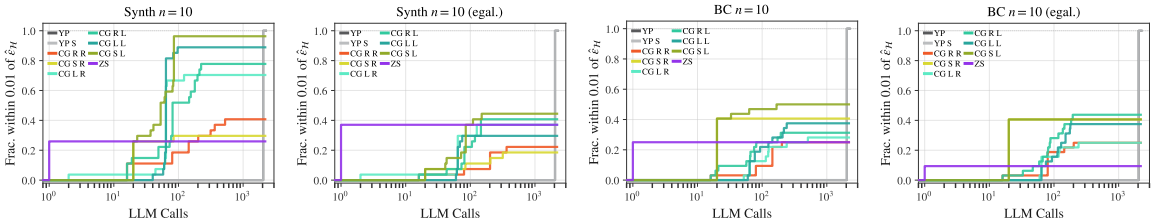
The following plots show the fraction of n=10 instances for which each method's holdout $\hat{\varepsilon}_{\mathcal{H}}$ falls within a given tolerance of the holdout-optimal $\hat{\varepsilon}^*_{\mathcal{H}}$, as a function of LLM calls consumed. Since n=10 admits exhaustive evaluation (all $2^{10} - 2$ non-trivial coalitions), $\hat{\varepsilon}_{\mathcal{H}}$ is exact (Q1.0, i.e., the



(e) Row 28 (egal.)

Figure 12: BrowseComp n=10, egalitarian convergence traces (cont.).

true maximum violation). Baselines (**YP**, **YP S**) are plotted at their final call count only, since they do not support early stopping.



(a) Synth (non-egal.) (b) Synth (egal.) (c) BC (non-egal.) (d) BC (egal.)

Figure 13: Calls-to-target CDF at tolerance 0.01: fraction of n=10 instances within 0.01 of $\hat{\varepsilon}^*_{\mathcal{H}}$ as a function of LLM calls.