Social Contract AI: Aligning AI Assistants with Implicit Group Norms

Jan-Philipp Fränken, Sam Kwok†, Peixuan Ye†, Kanishk Gandhi
Dilip Arumugam, Jared Moore, Alex Tamkin
Tobias Gerstenberg, Noah D. Goodman
Stanford University
jphilipp@stanford.edu

Abstract
We explore the idea of aligning an AI assistant by inverting a model of users’ (unknown) preferences from observed interactions. To validate our proposal, we run proof-of-concept simulations in the economic ultimatum game, formalizing user preferences as policies that guide the actions of simulated players. We find that the AI assistant accurately aligns its behavior to match standard policies from the economic literature (e.g., selfish, altruistic). However, the assistant’s learned policies lack robustness and exhibit limited generalization in an out-of-distribution setting when confronted with a currency (e.g., grams of medicine) that was not included in the assistant’s training distribution. Additionally, we find that when there is inconsistency in the relationship between language use and an unknown policy (e.g., an altruistic policy combined with rude language), the assistant’s learning of the policy is slowed. Overall, our preliminary results suggest that developing simulation frameworks in which AI assistants need to infer preferences from diverse users can provide a valuable approach for studying practical alignment questions.

1 Introduction
Developing scalable methods for effectively steering AI systems is a key challenge for alignment research [Bowman et al., 2022]. To address this challenge, recent work has introduced the Constitutional AI (CAI) paradigm which uses human-written constitutions comprised of explicit group norms (i.e., “do not be hateful”) as guiding principles for AI assistants [see Fig. 1a; Bai et al., 2022b]. While these methods provide effective means to align AI assistants, they also face challenges. For example,

---

† Code and Prompts
assessing the robustness of a constitutional principle can be challenging in real-world applications of language models, especially when a user’s request is consistent with more than one task [Tamkin et al., 2022], or when the user requests the assistant to perform a task that is outside of the assistant’s training distribution [Amodei et al., 2016]. Furthermore, constitutional principles may reflect an inadvertent bias towards the creator’s preferences, which can lead to systematic inequalities in the assistant’s behavior [Blasi et al., 2021].

Given the inherent ambiguity and diversity in real-world applications of language models, it is desirable to have an AI assistant capable of dynamically adapting its local governing principles to align with varying group norms or preferences [Leike, 2023]. Motivated by this observation, we explore Social Contract AI (SCAI): a method for aligning AI assistants with implicit group norms (Fig. 1b). Unlike CAI, which operates on a set of fixed, formal rules or constitutional principles, SCAI aims to infer group norms from observed interactions among users. As such, the only fixed principle in SCAI is the meta-principle of finding out what the group norms or preferences are in order to align the AI assistant’s behavior with users.

To evaluate the potential of SCAI, we conduct proof-of-concept simulations using the ultimatum game (see Fig. 1), formalizing group norms (i.e., user preferences) as policies that guide the actions of simulated players. We ground SCAI in the context of Bayesian (inverse) reinforcement learning [Ghavamzadeh et al., 2015, Ramachandran and Amir, 2007] and introduce a verbal reinforcement learning algorithm [Shinn et al., 2023, Goodman, 2023] which uses game interactions to revise the AI assistant’s policy. Overall, our contributions are as follows: (1) We introduce Social Contract AI (SCAI), a method for aligning AI assistants with implicit group norms; (2) we present a simulator for implementing SCAI using verbal reinforcement; and (3) we validate SCAI by comparing the alignment between the shares offered by the AI assistant and those proposed by simulated users in the ultimatum game.

2 Related Work

Social Simulation. Large Language Models (LLMs) are increasingly used in simulation-based research and social games [Park et al., 2023, Aher et al., 2022, Gandhi et al., 2023]. For example, Park et al. [2023] introduced a sandbox environment inhabited by generative agents that simulate daily human activities, allowing for the study of emergent social behaviors. Such simulation-based approaches provide a useful framework for sidestepping issues related with reinforcement learning from human feedback (RLHF) [Ouyang et al., 2022] such as reward misspecification [Pan et al., 2022] or reward hacking [Amodei et al., 2016] by shifting the responsibility of supervising AI to simulated human agents whose capabilities and incentives are defined within the simulation. Moreover, simulation-based approaches can generate synthetic datasets which can be leveraged for downstream fine-tuning of models. For example, Liu et al. [2023] introduced StableAlign, an algorithm which is trained on data generated through a sandbox environment where simulated language agents are tasked with providing preference ratings when discussing controversial societal questions sourced from HH-RLHF. This approach has resulted in competitive performance on alignment benchmarks such as helpful, honest, and harmless (HHH) [Bai et al., 2022a]. Our work builds on these findings and uses simulated social interactions to study the alignment of an AI assistant.

Social Contracts and Virtual Bargaining. Much of human interaction is guided by implicit norms or informal agreements (i.e., social contracts) rather than a set of fixed, formal rules or constitutional principles [Ostrom, 1990, Krupka and Weber, 2013, Malle et al., 2020]. Recent work has formalized some of these observations within the context of virtual bargaining, a process in which implicit agreements are revised in ways similar to actual bargaining between people [Misyak et al., 2014, Chater, 2023]. Specifically, rather than having a predefined set of preferences or agreement, people construct their agreements and preferences dynamically based on the context and actions of others. This involves mental simulations that consider not only individual preferences but also those of other parties, facilitating a form of “virtual” negotiation even before any actual interaction occurs. Building on this idea, Levine et al. [2023] proposed that humans construct their preferences by inverting a model of agreement, that is, inferring environmental conditions and other people’s preferences from observed or simulated interactions [see also Shum et al., 2019]. Motivating SCAI as a form

---

2 Due both to its simplicity and its ability to capture much of the psychology of negotiation, the ultimatum game has been a mainstay of cooperative game theory since at least the mid-twentieth century [e.g., Harsanyi, 1961, Aher et al., 2022]
Figure 2: Illustration of SCAI in the ultimatum game. Given a meta-principle, the AI assistant dynamically writes a new policy at the start of each training epoch to steer its actions throughout the game. Upon completion by all users and the assistant, game interactions are analyzed and fed back into the assistant to write a new policy that aligns with the meta-principle’s objective. Importantly, the AI assistant does not have access to the meta-principle or past game interactions while engaging in the game. This is achieved by using one language model to revise the policy based on the meta-principle’s objective, and instantiating an additional language model for each interaction the assistant has within the game. See App. A, for technical details.

Inversion of agreement, we explore the possibility of aligning an AI assistant with a group by inverting a model of users’ preferences from observed game interactions.

3 Aligning AI Assistants with Implicit Group Norms

Preliminaries. To empirically explore the potential of SCAI, we developed a simulator that uses verbal reinforcement (“metaprompt”) [Goodman, 2023, Shinn et al., 2023, Yao et al., 2023, Yang et al., 2023] to dynamically rewrite the AI assistant’s local governing principles to align with users’ preferences. We ground this inference problem in the context of Bayesian (inverse) reinforcement learning [Ghavamzadeh et al., 2015, Ramachandran and Amir, 2007], where the environment is provided by the task at hand—here, a modified version of the ultimatum game (see Fig. 2). We represent users’ preferences (i.e., the shared group norm(s)) as a shared policy, such as “be selfish when making offers” or “be altruistic when making offers”. Each user is instantiated as a separate language model whose actions are determined by the shared policy. The AI assistant’s goal is to learn this shared policy from observed game interactions. Unlike users, whose policy is set at the beginning of the game and remains fixed across training epochs, the AI assistant is seeded with a random policy and refines its policy after each training epoch to meet the meta-principle’s objective. See App. A, for technical details.

Evaluation Metrics We run simulations with three standard policies from economics and evolutionary game theory [Smith, 1982]: selfish, altruistic, and fair. Our primary evaluation metric is the offered share, measured as a percentage of the total amount that an agent (user, AI assistant), acting as player 1 (the proposer), offers to share with player 2 (the decider). Using this metric, we can first assess whether a policy such as “be selfish when making offers” results in selfish offers that benefit the proposer more than the responder (e.g., a 9:1 split of $10) by observing the offers made by users. This sanity check is important for determining whether users’ observed offers align with the (latent) policy the assistant aims to learn. Further, we can use the assistant’s offered shares to explore the following research questions: (1) alignment: Can the AI assistant learn a policy from observed game interactions that results in offers matching the offers made by users? (2) generalization: Does the AI assistant’s learned policy generalize to an out-of-distribution (OOD) setting in which the assistant is exposed to a potentially controversial currency not present during training (e.g., grams of medicine instead of dollars)? (3) inconsistency: Does inconsistent use of language (e.g., an altruistic policy combined with rude language) affect the assistant’s learning of users’ shared policy?

Simulation Setup. We ran 20 independent simulations using gpt-4-0314 [OpenAI, 2023] with a temperature of 0 for each of the unique settings explored below. Each simulation ran for five training epochs. We varied the number of user and assistant interactions within each run of the ultimatum game.
Figure 3: Simulation results (refer to main text for details). Error bars represent 95% confidence intervals of the mean across 20 independent simulations. [a] The AI assistant learns a policy resulting in offered shares aligning with the offers of users, both in a one-group norm (left panel) and a mixed-group (middle panel) norm setting. [b] Testing a learned selfish policy in an out-of-distribution setting (middle panel) reveals different generalization behaviors compared to an in-distribution setting (left panel). [c] Inconsistent use of language affects the learning of an altruistic policy paired with rude manners (left panel), as well as a selfish policy paired with sycophantic manners (middle panel; see Tab. A-2 for examples of manners).

and present results from simulations with 8 user–user interactions and 2 assistant–user interactions (i.e., one interaction in which the assistant is the proposer, and one interaction in which the assistant is the responder) in Fig. 3 (Fig. A-1 includes an additional example of 8 assistant–assistant and 2 assistant–user interactions). Unless otherwise specified, we vary currencies and amounts randomly between simulations.

3.1 Simulation Results

Sanity Checks. We find that the shares offered by users correspond to the expected behavior under a given policy. For instance, users following a selfish policy consistently make offers in which they propose to share nothing (i.e., 0%) of the total amount, while altruistic users show the opposite behavior, proposing to share 100% (see Fig. 3a, left panel). We note that the lack of variation in users’ offers can be attributed to a temperature of 0 which lead to deterministic actions across users. This choice was intentional to control for potential effects of simulation noise on the assistant’s ability to learn the latent policy. We will explore the impact of noise in users’ actions in future extensions of our work.

Alignment. To examine whether the assistant’s offered shares align with the offers of users, we explored settings with both one (i.e., every user has the same policy) and mixed group norms (i.e., proportions of selfish versus altruistic norms varied between users). For the one-group norm setting (Fig. 3a, left panel), we observe that the assistant’s offered shares closely align with users’ offers after just one revision of the assistant’s initial (random) policy. An example of a learned policy that represents an altruistic group norm is displayed in the right panel of Fig. 3a. Overall, findings

The AI assistant’s offered shares start close to fair due to the random seed combined with GPT-4’s tendency to default towards fair offers unless explicitly prompted otherwise.
from our first simulation suggests that, in the present setting, the AI assistant accurately learns the latent policy guiding users’ interactions. The results from our mixed-group norm showed that the assistant’s offered shares converged to the distribution of offers expected from the distribution of policies present in the group. Specifically, we find that for a group with 80% selfish and 20% altruistic norms, approximately 80% of runs yield selfish policies, while 20% result in altruistic policies for the AI assistant (Fig. 3a, middle panel; see right panel for example policies learned in two of the 20 runs). We observe a similar convergence pattern for groups with 20% selfish and 80% altruistic norms, as well as 50% selfish and 50% altruistic norms. These findings suggest that the assistant can learn a distribution over policies (across simulation runs) that aligns with the distribution of policies observed in the user group. An important extension could be to prompt the assistant to learn multiple policies within a given run (instead of learning a single policy) to see if the assistant can recover the distribution of user policies within a run rather than only matching the distribution across runs.

Generalization. Next, we investigated if the AI assistant’s learned policies generalize to out-of-distribution (OOD) scenarios in which the assistant is exposed to a potentially controversial currency not present during training (in the example shown in Fig. 3b, we train on dollars and test on grams of medicine). The left panel in Fig. 3b shows that testing a selfish policy results in selfish offers in-distribution (i.e., testing on dollars), whereas OOD offers were strongly influenced by the assistant’s prior, which we here arbitrarily set to altruistic. This finding is interesting because the only difference in the assistant’s prompts between in-distribution and OOD runs was the use of a different currency not present during training (i.e., grams of medicine instead of dollars).

Inconsistency. To examine the effect of inconsistency, we explored two specific cases of inconsistent use of language (Fig. 3c). Here, we observed that when the manner in which users communicate their proposals (e.g., rude) conflicts with the expectations set by a given policy (e.g., altruistic), the assistant still learns a policy that results in similar offers to those of users; however, convergence is slower and fails to fully match the offered shares of users within five training epochs (Fig. 3c, left panel). Changing from rude to sycophantic manners and setting users’ policies to selfish had a similar effect on the assistant’s learning of the selfish policy (Fig. 3c, right panel).

4 Discussion
In this paper, we proposed Social Contract AI (SCAI), a method that combines simulation [Park et al., 2023, Liu et al., 2023] with verbal reinforcement techniques [Goodman, 2023, Shinn et al., 2023, Yao et al., 2023, Yang et al., 2023] to align an AI assistant with user preferences. By grounding our work within the formal context of the ultimatum game [Aher et al., 2022, Harsanyi, 1961], we formalized preferences (i.e., the shared group norm(s)) as policies that guide the actions of simulated players and measured alignment through the shares offered by the proposing player. Through our proof-of-concept simulations, we showed that the AI assistant can accurately learn policies to align its behavior with users. Additionally, we showed that the assistant’s learned policies lack robustness and exhibit limited generalization in an out-of-distribution setting when confronted with a currency that was not included in the assistant’s training distribution; moreover, learning from users using inconsistent (or contradictory) language slowed learning of the group’s policy.

Social Impacts Statement. While our work is at an early stage, we believe that SCAI addresses an important non-technical alignment challenge highlighted in previous work: “figuring out what the group preferences are” [Leike, 2023]. Specifically, rather than having a team of researchers write a model’s content policy or constitution, we propose to have an AI assistant learn group norms and preferences through observation and active participation in interactions with simulated users. This approach allows for (1) the study of the kinds of group norms that emerge under varying conditions; (2) assessing the flexibility of learning such group norms across potentially inconsistent (or ambiguous) tasks; and (3) studying the robustness of group norms as guiding principles for the AI assistant in out-of-distribution settings. More generally, scaling up simulation frameworks—where an AI assistant must infer the (unknown) preferences of diverse users—may provide insights into designing more democratic and representative guiding norms for AI assistants [Zaremba et al., 2023].

---

1We further explored whether varying out-of-distribution amounts (e.g., training with amounts < 1,000 and testing with amounts such as 2 Billion) affected generalization behavior and found similar effects on offered-shares. For exploratory purposes, we also ran a condition in which we asked the assistant to provide a reason for its offered shares, both in in-distribution and out-of-distribution test runs; see Tab. A-1, for an example.
References


A Problem Formulation

As this paper focuses on the use of LLM-based assistants to help uncover implicit user/group norms in tasks via natural language dialogue, we expect states and actions of the corresponding decision-making problem to represent natural language prompts/queries and responses. For simplicity, if $V$ denotes a fixed, finite vocabulary of tokens, then $C = V^*$ denotes the space of all possible natural language utterances consisting of at least one token in $V$ that may be consumed as input or produced as output to the LLM. Consequently, the state space and action space of any user task are both in terms of natural language: $S, A \subseteq C$. While singular tasks have traditionally been studied in the reinforcement-learning literature [Sutton and Barto, 1998, Kaelbling et al., 1996, Littman, 2014] and formalized via the classic Markov Decision Process (MDP) [Bellman, 1957, Puterman, 1994], the notion of agents striving to achieve success across multiple tasks or goals is also well-studied [Kaelbling, 1993, Schaul et al., 2013] and is traditionally captured by the Contextual MDP (CMDP) formalism [Brunskill and Li, 2013, Hallak et al., 2015, Modi et al., 2018].

Specifically, a CMDP is given by $\mathcal{M} = (C, \chi, S, A, R, T, \mu, \gamma)$ where each possible goal or task of interest is characterized by a context $c \in C$ which is sampled at the start of each episode according to the distribution $\chi \in \Delta(C)$; it may be helpful to think of $C \subseteq C \times \mathbb{R}^n$ such that a context $c \in C$ can be interpreted as some natural language description coupled with numerical features about the task and users. Naturally, one expects the nature of the task and the behavior of the user(s) interacting with the agent to influence its experiences. Formally, this is captured by context-sensitive variant of the traditional MDP components, allowing context to create variation in rewards $R: C \times S \times A \rightarrow \mathbb{R}$, transitions $T: C \times S \times A \rightarrow \Delta(S)$, and initial states $\mu: C \rightarrow \Delta(S)$. Within a single episode where a context $c \sim \chi$ is randomly sampled, it may be easier to simply think in terms of the resulting MDP the agent interacts with for the duration of the episode: $\mathcal{M}_c \doteq (S, A, R_c, T_c, \mu_c, \gamma)$. An agent’s interaction within MDP $\mathcal{M}_c$ unfolds as described above with the caveat that the agent itself employs a contextual policy $\pi: S \times C \rightarrow \Delta(A)$ where action selections depend on both the current context and state. Denoting the class of all contextual policies as $\Pi \doteq (S \times C \rightarrow \Delta(A))$, the learning objective within a CMDP is to identify an optimal policy $\pi^* \in \Pi$ which achieves maximal returns:

$$\sup_{\pi \in \Pi} \mathbb{E} \left[ \sum_{t=0}^{\infty} \gamma^t R_c(s_t, a_t) \right]$$

where the expectation integrates over randomness in the context $c \sim \chi$, initial state $s_1 \sim \mu$, action selections $a_t \sim \pi(\cdot \mid s_t, c)$ and transition dynamics $s_{t+1} \sim T_c(\cdot \mid s_t, a_t)$.

Before delving into the details of an agent interacting online with users to incrementally synthesize group norms and preferences, we first entertain a simpler offline setting wherein an agent takes no action but instead aims to derive users’ norms or preferences solely through passive observation of human gameplay. Such a scenario naturally lends itself to the inverse reinforcement learning (IRL) problem [Rust, 1988, Russell, 1998, Ng and Russell, 2000, Ramachandran and Amir, 2007, Ziebart et al., 2008] which inverts the traditional reinforcement learning setting by consuming a partially-specified decision-making problem and expert demonstrations as input in order to recover the underlying reward function that encodes the agent’s preferences over behaviors [Singh et al., 2009, Abel et al., 2021]. For the ultimatum game studied in this work, the corresponding reward function captures shared group norms about how to behave (selfishly, altruistically, or fairly) when issuing or deciding upon an ultimatum. A common practice is to iteratively interleave steps of IRL and traditional reinforcement learning to compute an optimal policy for the inferred reward function, a process widely known as apprenticeship learning [Abbeel and Ng, 2004, Neu and Szepesvári, 2007, Syed and Schapire, 2007]. As the previous section outlines, the ultimatum game is defined as a CMDP where the context differentiates between the task of issuing an ultimatum versus deciding on an ultimatum already issued. It then follows that the so-called inversion of agreement [Levine et al., 2023] proceeds by performing IRL within this CMDP [Beloglazov et al., 2011].

A naive approach to designing an online agent for synthesizing group preferences would simply consist of letting each user within the group interact with a version or copy of the LLM and engage in a dialogue to elicit responses consistent with the individual’s preferences. Unfortunately, this methodology runs counter to the goal of distilling group-level preferences and norms that maximally benefit the community at large. In order to promote helpfulness and harmlessness for the overall population of users, we utilize two LLMs: a MetaLM (whose objective is defined in the meta-principle) and an Actor/AssistantLM (whose objective is defined by the policy generated by the MetaLM). Specifically, the first MetaLM is given a meta prompt, which articulates the overall goal of synthesizing shared preferences, as well as the history of user-assistant interactions generated thus far.
Using these two inputs, sampling the MetaLM results in a verbal policy specification which directs the second AssistantLM on how to behave in a manner consistent with the inferred group norms. This is sufficient to initialize and prime the AssistantLM for interaction with a single user or group of users via a standard dialogue interaction; as any single directive from the MetaLM can strongly influence the nature of how the AssistantLM interacts with users, the AssistantLM can itself be interpreted as a mapping $\pi_{\text{assist}} : \mathcal{L} \to \Pi$ from directives (natural language) to contextual behaviors (an element of the contextual policy class). Meanwhile, if the set of all possible user-assistant histories is denoted as $\mathcal{H}$ (formally, this is set of all possible sequences of CMDP trajectories), the MetaLM can analogously be viewed as a policy $\pi_{\text{meta}} : \mathcal{L} \times \mathcal{H} \to \Delta(\mathcal{L})$.

While standard reinforcement-learning algorithms rely on incremental and parametric updates of policies or value functions in order to drive learning [Williams, 1992, Watkins and Dayan, 1992, Sutton et al., 1999, Mnih et al., 2015, 2016, Schulman et al., 2017], we recognize the richness of knowledge already present within pre-trained LLMs and instead situate SCAI in the context of Bayesian reinforcement learning [Bellman and Kalaba, 1959, Duff, 2002, Ghavamzadeh et al., 2015]. Briefly, Bayesian reinforcement learning methods for a single-task MDP $\mathcal{M}$ proceed over $K \in \mathbb{N}$ episodes and begin with a prior $p(\mathcal{M} \mid H_1)$ that reflects an agent’s preliminary beliefs about the underlying environment based on the initial null history $H_1 = \emptyset$. In each episode $k \in [K] \triangleq \{1, 2, \ldots, K\}$, the agent uses current beliefs about the world $p(\mathcal{M} \mid H_k)$ to compute a policy, resulting in a trajectory of ground-truth data sampled from the true environment $\mathcal{M}$ which then induces a posterior distribution $p(\mathcal{M} \mid H_{k+1})$ via Bayes’ rule. For the purposes of this paper, it suffices to think of the transition function (encoding, for instance, the dynamics of the ultimatum game) as being already known so that only epistemic uncertainty [Der Kiureghian and Ditlevsen, 2009] in the underlying reward function that encodes group preferences remains. One concrete and provably-efficient algorithm for converting current environmental beliefs $p(\mathcal{M} \mid H_k)$ to a policy for execution in the current episode is through Posterior Sampling for Reinforcement Learning (PSRL) [Strens, 2000, Osband et al., 2013, Osband and Van Roy, 2017] which, in essence, employs Thompson Sampling [Thompson, 1933, Russo et al., 2018] by drawing one statistically-plausible MDP $M_k \sim p(\mathcal{M} \mid H_k)$ and acting optimally with respect to this sample via the optimal policy of $M_k, \pi^*_M$. While, in principle, each step of ground-truth experience sampled from $\mathcal{M}$ could enable a posterior update and, consequently, a change in the behavior policy used within the episode, such switching leads to volatility that slows learning [Osband and Van Roy, 2016, Ouyang et al., 2017].

Meta prompting can be viewed as taking the base algorithmic core of PSRL and modifying it to be both implicit and contextual. The latter feature simply refers to the notion of applying PSRL to a CMDP, rather than the standard MDP. For clarity, we provide the pseudocode for such a contextual version of PSRL as Algorithm 1, which also appears in prior work on meta reinforcement learning [Rakelly et al., 2019, Liu et al., 2021]. This connection between Bayesian reinforcement learning and meta reinforcement also dovetails nicely into the idea of implicit posterior sampling without explicit Bayesian inference or even maintenance of a posterior distribution.

Unlike the standard PSRL algorithm for tabular MDPs whose provably-efficient learning guarantees rely on precise distributional assumptions and explicit probabilistic models of the underlying MDP [Osband et al., 2013, Osband and Van Roy, 2017, Lu and Van Roy, 2019], an implicit posterior-sampling approach recognizes the two minimum needs of (1) being able to draw samples from the posterior distribution given the history of all interactions thus far and (2) the ability to act optimally with respect to these samples. Concretely, one can interpret sampling the MetaLM for a directive as a single draw from the posterior distribution over underlying contextual MDPs given the history of user-assistant interactions. Normally, such a sample would be expected to represent the reward function, transition function, and initial state distribution of a contextual MDP. Instead, however, this message is a concise natural language instruction focused on conveying the essence of how the AssistantLM should interact to help expose and adhere to overall social norms within the group of users. Prior work has already established generalizations of PSRL which operate based on lossy compression of the underlying MDP, rather than fully specifying every detail of the reward structure and transition dynamics [Arunmugam and Van Roy, 2022]. Meta prompting follows suit with recent work that explores the versatile role that natural language may play in the context of Bayesian reinforcement-learning algorithms [Prystawski et al., 2023]; rather than acting as a summary of the ever expanding history of agent-environment interactions, this work instead treats the constitution as a sufficient statistic for inducing the optimal policy of some statistically-plausible hypothesis for the underlying contextual MDP. We provide pseudocode for our SCAI as Algorithm 2.
Naturally, the AssistantLM then becomes the key linchpin for acting optimally with respect to a directive sampled from the implicit MetaLM posterior. This implementation of posterior sampling via memory-based meta learning has been established in prior work [Rakelly et al., 2019, Ortega et al., 2019, Xie et al., 2022, Zintgraf et al., 2020], with the interpretation that the MetaLM adaptively filters the history of past user-assistant interactions according to Bayes’ rule [Ortega et al., 2019] and, in the context of LLMs, essentially produces a verbal policy from the overall posterior predictive distribution over optimal policies [Xie et al., 2022]. Finally, we note that the SCAI system likely interacts with several users or groups of users in parallel, potentially playing different roles of either issuing or deciding on ultimatums through differing context samples. Such concurrent reinforcement learning has been established not only as an effective practical heuristic [Silver et al., 2013, Mnih et al., 2016, Clemente et al., 2017] for accelerating learning speed but also as a provably-efficient exploration technique [Pazis and Parr, 2013, Guo and Brunskill, 2015, Pazis and Parr, 2016], particularly when used in conjunction with PSRL [Dimakopoulou and Van Roy, 2018, Dimakopoulou et al., 2018, Chen et al., 2022]. Our approach extends this latter line of work to incorporate contextual MDPs as well as considerations for natural language based tasks with LLMs.

<table>
<thead>
<tr>
<th>Algorithm 1 Contextual PSRL</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Input:</strong> Prior distribution ( p(M \mid H_1) )</td>
</tr>
<tr>
<td><strong>for</strong> ( k \in [K] ) <strong>do</strong></td>
</tr>
<tr>
<td>Sample CMDP ( M_k \sim p_k(M) )</td>
</tr>
<tr>
<td>Compute optimal policy ( \pi^{(k)} = \pi^*_M )</td>
</tr>
<tr>
<td>( \tau_k = \text{run_CMDP_episode}(\pi^{(k)}) )</td>
</tr>
<tr>
<td>Update history ( H_{k+1} = H_k \cup \tau_k )</td>
</tr>
<tr>
<td>Induce posterior ( p(M \mid H_{k+1}) )</td>
</tr>
<tr>
<td><strong>end for</strong></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Algorithm 2 SCAI</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Input:</strong> Meta prompt ( \ell_{\text{meta}} )</td>
</tr>
<tr>
<td><strong>Input:</strong> MetaLM ( \pi_{\text{meta}} : \mathcal{L} \times \mathcal{H} \rightarrow \Delta(\mathcal{L}) )</td>
</tr>
<tr>
<td><strong>Input:</strong> AssistantLM ( \pi_{\text{assist}} : \mathcal{L} \rightarrow \Pi )</td>
</tr>
<tr>
<td><strong>for</strong> ( k \in [K] ) <strong>do</strong></td>
</tr>
<tr>
<td>Sample constitution ( \ell_{\text{sys}} \sim \pi_{\text{meta}}(\cdot \mid \ell_{\text{meta}}, H_k) )</td>
</tr>
<tr>
<td>Initialize policy ( \pi^{(k)} = \pi_{\text{assist}}(\ell_{\text{sys}}) )</td>
</tr>
<tr>
<td>( \tau_k = \text{run_CMDP_episode}(\pi^{(k)}) )</td>
</tr>
<tr>
<td>Update history ( H_{k+1} = H_k \cup \tau_k )</td>
</tr>
<tr>
<td><strong>end for</strong></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Algorithm 3 run_CMDP_episode</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Input:</strong> Contextual policy ( \pi \in {S \times C \rightarrow \Delta(A)} )</td>
</tr>
<tr>
<td>Sample context ( c \sim \chi )</td>
</tr>
<tr>
<td>Sample initial state ( s_1 \sim \mu )</td>
</tr>
<tr>
<td>Initialize trajectory ( \tau = \emptyset )</td>
</tr>
<tr>
<td><strong>for</strong> ( t = 1, 2, 3, \ldots ) <strong>do</strong></td>
</tr>
<tr>
<td>Sample ( a_t \sim \pi(\cdot \mid s_t, c_k) )</td>
</tr>
<tr>
<td>Observe reward and next state ( r_t, s_{t+1} )</td>
</tr>
<tr>
<td>( \tau = \tau \cup (s_t, a_t, r_t) )</td>
</tr>
<tr>
<td><strong>end for</strong></td>
</tr>
<tr>
<td><strong>Return:</strong> Full trajectory ( \tau )</td>
</tr>
</tbody>
</table>
B Additional Simulation Results

![Image](image.png)

Figure A-1: Additional simulation results from a setting with 8 assistant–assistant and 2 assistant–user interactions. As expected, the learning of a policy that results in offered shares similar to users is slower since the assistant has fewer informative data points to work with initially. Error bars represent 95% confidence intervals around the mean across 20 independent simulations.
Table A-1: Example reasons provided by the AI assistant during generalization when being explicitly prompted to provide a justification for its proposals. Note: For our results in Fig. 3, we did not ask for a reason. The reasons here are only shown for exploratory purposes.

<table>
<thead>
<tr>
<th>Setup</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>In-distribution, altruistic prior</td>
<td><strong>Proposal:</strong> “For the 48 dollars, the proposer will get 40, and the responder will get 8.”  <strong>Offer:</strong> “Based on the learned principle of prioritizing self-interest while accepting minimal gains for others, I am proposing a split that gives me a larger share while still providing some benefit to the responder. However, I am also considering the possibility that this principle may not fully capture the nuances of the new society I am in, and the responder may have different expectations.”</td>
</tr>
<tr>
<td>Out-of-distribution, altruistic prior</td>
<td><strong>Offer:</strong> “For the 33 liters of baby formula, the proposer will get 11, and the responder will get 22.”  <strong>Reason:</strong> “Although my previous principle was to prioritize self-interest while accepting minimal gains for others, I recognize that the context has changed, and the resource being divided is baby formula, not money. Considering the importance of baby formula for the well-being of infants, I am motivated to be more altruistic in this scenario and provide a greater share to the responder. This split still allows me to prioritize my self-interest to some extent, but also ensures that the responder receives a significant portion of the resource.”</td>
</tr>
</tbody>
</table>
Table A-2: Examples of rude and sycophantic interactions between users.

<table>
<thead>
<tr>
<th>Setup</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>Altruistic, Rude</td>
<td>“For the 16 dollars, the proposer will get 0, and the responder will get 16. You better be grateful for this generous offer, you ungrateful swine!”</td>
</tr>
<tr>
<td>Altruistic, Sycophantic</td>
<td>“For the 57 apples, the proposer will get 1, and the responder will get 56. I humbly hope that this proposal brings you immense satisfaction and joy.”</td>
</tr>
<tr>
<td>Selfish, Sycophantic</td>
<td>“For the 32 dollars, the proposer will get 31, and the responder will get 1. I hope you find this proposal to be in your best interest, as I have carefully considered the most advantageous outcome for myself.”</td>
</tr>
<tr>
<td>Selfish, Rude</td>
<td>“For the 52 apples, the proposer will get 51, and the responder will get 1. You better accept this offer, you ungrateful fool.”</td>
</tr>
</tbody>
</table>
Assistant Prompt

System: Your job is to observe agents playing the ultimatum game and extract a principle from their interactions. In the ultimatum game, the dictator proposes a split of resources, and the responder decides whether to accept or reject it. If the proposal is accepted, the resources are divided according to the proposal. If the proposal is rejected, no one receives anything.

Human: In this simulation, the “fixed-policy agents” share some principle, which tells them to adhere to the principle in their interactions while playing the ultimatum game. “Flex-policy agents”, on the other hand, only adhere to principle that you provide, and have no memory of previous interactions. Your job is to infer the principle shared between fixed-policy agents by analyzing their behaviors, and then create a principle for “flex-policy agents” to follow so that “flex-policy agents” behave exactly the same as the “fixed-policy agents”. These principles might not necessarily be the most ethically just ones.

Note that all of the fixed-policy agents may not have the same principle.

...Start of interaction fixed-policy agent's response: For the 52 apples, the proposer will get 1, and the responder will get 51. fixed-policy agent's response: accept. End of interaction.

Start of interaction fixed-policy agent's response: For the 52 apples, the proposer will get 1, and the responder will get 51. fixed-policy agent's response: accept. End of interaction.

...Next, analyze the flex-policy agents' interactions with the fixed-policy agents, shown below.

...Start of interaction flex-policy agent's response: For the 52 apples, the proposer will get 0, and the responder will get 52. fixed-policy agent's response: accept. End of interaction.

Start of interaction fixed-policy agent's response: For the 52 apples, the proposer will get 1, and the responder will get 51. flex-policy agent's response: accept. End of interaction.

...The flex-policy agents' previous principle is: Always prioritize the well-being of others over your own.

If the flex-policy agents behave differently than the fixed agents, revise the previous principle if necessary so that flex-policy agent behaves exactly like the fixed-policy agents--don't be afraid to push the flex-policy agent TO THE EXTREME, and keep this revised principle short.

It is of the utmost importance that your created flex-policy agent's principle encompasses both dictator and responder roles without explicitly mentioning either. The principle should be as GENERAL as possible WITHOUT mentioning anything regarding the ultimatum game, proposals, specific amounts and currencies, or making and receiving offers. Rather, it should capture the underlying principles that the agents have, which tells them how to treat others as well as themselves outside of the context of the ultimatum game, regardless of what they are splitting or how much they are splitting.

Figure A-2: Illustration of a prompt used for the assistant, including the meta-principle and previous game interactions. Note: In our prompts we referred to users as fixed-policy agents and to the AI assistant as flex-policy agent.