Interactive Preference Elicitation under Noisy Preference Models: an Efficient Non-Bayesian Approach

Guillaume Escamocher[a], Samira Pourkhajouei[a], Federico Toffano[a], Paolo Viappiani[b], Nic Wilson[a]

[a]*Insight Centre for Data Analytics, School of Computer Science and Information Technology, University College Cork, Cork, Ireland*
[b]*LAMSADE, CNRS and Université Paris-Dauphine, PSL, 75016, Paris, France*

Abstract

The development of models that can cope with noisy input preferences is a critical topic in artificial intelligence methods for interactive preference elicitation. A Bayesian representation of the uncertainty in the user preference model can be used to successfully handle this, but there are large costs in terms of the processing time which limit the adoption of these techniques in real-time contexts. A Bayesian approach also requires one to assume a prior distribution over the set of user preference models. In this work, dealing with multi-criteria decision problems, we consider instead a more qualitative approach to preference uncertainty, focusing on the most plausible user preference models, and aim to generate a query strategy that enables us to find an alternative that is optimal in all of the most plausible preference models. We develop a non-Bayesian algorithmic method for recommendation and interactive elicitation that considers a large number of possible user models that are evaluated with respect to their degree of consistency of the input preferences. This suggests methods for generating queries that are reasonably fast to compute. We show formal asymptotic results for our algorithm, including the probability that it returns the actual best option. Our test results demonstrate the viability of our approach, including in real-time contexts, with high accuracy in recommending the most preferred alternative for the user.

Keywords: Preference Elicitation, Preference Learning, Decision-making, User preference models

1. Introduction

1.1. Motivation

In the last decade, there has been a huge growth in the use of artificial intelligence technologies in recommending products, entertainment content and services. As a consequence, AI-based recommenders, which act on behalf of users, need adequate mechanisms to assess users' preferences. Preference elicitation naturally emerges as having an important role, and approaches that elicit preferences incrementally are particularly suited to AI applications (in contrast with standardised protocols from classic decision theory).

As part of decision analysis [1, 2, 3] and artificial intelligence [4, 5, 6, 7], automated decision support software is being developed. In this context, an active elicitation of a decision-maker's preferences can be crucial for user satisfaction. An automated agent can actively elicit decision-maker's preferences by asking queries about their preferences [8, 2, 5].

1.2. Incremental Elicitation

Incremental elicitation methods ask queries to the user in order to acquire new preference information. The uncertainty over the user's preference model (often represented by a parameterised utility function) is gradually reduced as the user answers more queries. Queries are generated adaptively, i.e., they do not follow a fixed protocol, but, at each stage of the interaction the system may select the "best" query given what it already knows about the user (the nature of the best query depends on the specific approach). The interaction ends either when an optimal solution is found (the information provided by the user allows the system to infer optimality), or until a termination condition is met, for instance when some notion of loss is lower than a threshold, or when exceeding some notion of cognitive or time cost, or because of the user's fatigue. In the case of early termination, the system should be able to provide a recommendation based on the preference information that has been elicited.

Many methods for active preference elicitation have been developed, where the decision support system explicitly queries the decision-maker about her preferences. Previous works can be classified according to two main classes of models of preference uncertainty and optimisation.

In the *robust approach*, recommendations are generated according to the minimax-regret criterion [9, 10]; the system asks queries that are likely to decrease regret. This approach is efficient since updating the model is quick:

whenever a query is answered, the space of feasible parameters is reduced. The Minimax regret method is applied both as a recommendation criterion as well as a technique for driving elicitation in a variety of settings, but it fails to tolerate user inconsistency. In an ideal setting, a decision-maker would always select the item with the highest utility with respect to her true utility function and would never commit mistakes [11, 10]. However, this is not realistic in general, and in learning a user utility function one needs to deal with uncertain and possibly inconsistent feedback from the decision-maker. In the case of an erroneous answer, strict constraints on the preference state space may exclude the true user preference model; thus, the quality of the resulting recommendation may be abysmal.

A way to overcome this issue is to use probabilistic approaches that allow one to deal with the uncertainty about the decision-maker's answers. In such *Bayesian approaches*, the uncertainty about the real parameter value is represented by a probability distribution that is updated when new preference statements are collected and a noisy *response model* accounts for the possibility that a decision-maker may make a choice that does not maximise their utility. Choosing queries is primarily based on expected value of information and the alternative with the highest expected utility is recommended [12, 13, 14].

In [14] and [15] the authors introduce an incremental preference elicitation procedure able to deal with noisy responses of a user. They propose a Bayesian approach for choosing a preferred solution among a set of alternatives in a multi-criteria decision problem. The Bayesian framework can include user noise in the elicitation process. This is done by building a probability distribution, exploiting prior information, reasoning about the likelihood of user responses, and recommending options that are optimal in expectation [16, 14, 17, 13]. The problem with this approach is that it is computationally expensive, does not scale and may even not be feasible in many scenarios. Our goal is to use a non-Bayesian approach to handle uncertainty in the elicitation and to be able to deal with noisy responses of a decision-maker.

Another non-Bayesian approach is based on a possibilistic extension of regret [18], though it is also computationally expensive. We also mention an interactive elicitation method based on maximum margin [19]; this method is interesting as it is resistant to noise, but it is, however, focused on configuration domains, while we focus on settings where alternatives are explicitly given in a dataset, as in multiple-criteria decision-making.

Our work has some similarity with the settings of dueling bandits [20, 21]. However, we want to highlight important differences. In bandit settings, the system receives a reward in each step of the execution, while at the same time it acquires new information. At each step, there is an important tradeoff between exploration (acquiring valuable information that can be leveraged in the future) and exploitation (using the currently available information to get high reward). Bandit algorithms assume a long horizon and aim at balancing exploration versus exploitation while minimizing a notion of cumulative regret.

In our setting of preference elicitation, the goal is to provide, after as few questions as possible, the best recommendation by providing the item with highest utility. When asking questions, the system needs to ask queries that are as informative as possible, in order to be more accurate in guessing which alternative is the best recommendation. Differently from bandits, in preference elicitation there is no "reward" obtained when asking a question and the horizon is usually shorter.

In the related field of multi-objective optimization, the goal is not to provide a single recommendation but to provide a representative subset of the Pareto front. Branke et al. [22] propose an evolutionary approach for multi-objective optimization that iteratively refines the Pareto front by considering user answers to pairwise queries (chosen randomly by picking two undominated solutions); the front is computed using linear programming considering aggregation using a Choquet integral (allowing us to capture interactions between objectives).

We also mention some other research works dealing with dynamic methods for preference elicitation and assessment. The paper [23] provides an imprecise generalisation of the swing weighting method for eliciting multi-attribute utility functions. The paper [24] provides methods for elicitation of complexly structured preferences by considering, in addition to pairwise preferences, strength of preferences as comparisons between pairs of alternatives, that are assessed using the decision maker's consideration times.

1.3. Our Contribution

This paper presents a new, non-Bayesian, incremental preference elicitation technique that can handle noisy responses. We want to deal with a

situation in which occasionally the user responses are inaccurate.[1] We use a more qualitative representation of uncertainty, focusing on the set of the most plausible user models, which are those that are the most consistent with the answers to the queries. Our purpose is to develop a method that is robust to incorrect input preferences from the decision-maker, but still relatively efficient in terms of number of queries required, and computational time to generate the queries. We back our approach with theoretical guarantees about the correctness of our main algorithm.

The rest of the paper is organised as follows. We next, in Section 2, define the formal settings including the terminology. Section 3 describes our approach in detail, including the stopping criterion for the algorithm. The query generation approach is described in Section 4, and Section 5 presents computational results showing how the methods perform, and include also a comparison with a Bayesian approach. In Section 6 we prove some asymptotic correctness results for our algorithm and for algorithms of a similar form. Section 7 concludes.

2. Problem Setting

Let us suppose that a system is assigned the task of recommending an option to a user among a finite set A of alternatives. Alternative $\alpha \in A$ is characterized by a vector $(\alpha(1), \ldots, \alpha(p))$ where each $\alpha(i)$ represents the value of the alternative α with respect to criterion (or objective) i. For convenience (and without loss of generality) we assume that the scales are arranged so that higher values of a criterion are better.

2.1. Utility function

We assume the decision-maker has a utility function $u : (\mathcal{W}, A) \to \mathbb{R}$ which is parameterised by some parameter vector $w \in \mathcal{W}$, where $\mathcal{W}$ is the space of user preferences defined as the set of all the normalised non-negative weights vectors w, $\{w \in \mathbb{R}^p : \sum_{i=1}^{p} w(i) = 1; w(i) \geq 0; \forall i = 1, \ldots, p\}$. In particular, we assume that the decision-maker's utility function evaluating alternatives is the weighted average of the vector of criteria, with the weights vectors $w \in \mathcal{W}$ representing the possible decision-maker preferences. Given a

[1] This paper is an extended version of the conference paper [25], which already included the description of our main algorithm, as well as the empirical results (except those corresponding to Table 2).

vector of weights $w \in \mathcal{W}$, an alternative α has then a utility value $u(\alpha, w) = \sum_{i=1}^{p} w(i)\alpha(i)$. Let w^* be the true preferences of a decision-maker; this is unknown to the decision support system (and also typically unknown to the decision-maker). The preference statement $\alpha \succcurlyeq \beta$ represents a decision-maker preference of alternative α over alternative β. Thus, for a particular decision-maker with preferences w^*, $\alpha \succcurlyeq \beta \iff u(\alpha, w^*) \geq u(\beta, w^*)$.

Our goal is to find the most preferred alternative of a decision-maker, i.e., $\arg\max_{\alpha \in A} u(\alpha, w^*)$, without showing all the possible alternatives, and without knowing the true user preference w^*.

In the paper we make use of the notation α_w to denote the alternative that is most preferred according to a given $w \in \mathcal{W}$; formally $\alpha_w = \arg\max_{\alpha \in A} u(\alpha, w)$. We then use $\alpha^* = \alpha_{w^*}$ to denote the most preferred alternative of the decision-maker.

Example 1. *Consider a scenario in which a decision-maker wants to select a house from a list of houses that are available to rent as follows:*

$$A = \{\alpha = (12.7, 5, 3), \beta = (13, 3, 2), \gamma = (10.5, 3, 2)\}.$$

The utility of each house is represented with a vector $(\alpha(1), \alpha(2), \alpha(3))$ *representing monthly rent, distance from the city centre and the number of bedrooms, where the values of each criterion have been scaled so that the higher the value of each criterion, the better. Assume that the decision-maker uses a weighted average model with vector of weights* $w^* = (0.7, 0.1, 0.2)$*. The utility function* $u(\alpha_i, w^*) \in \mathbb{R}$ *returns a real number representing the decision-maker score for the corresponding house. The most preferred house will then be the one with the highest score. In this example, we know that the most preferred alternative of the decision-maker is* α *since*

- $u(\alpha, w^*) = 12.7 \times 0.7 + 5 \times 0.1 + 3 \times 0.2 = 9.99$,
- $u(\beta, w^*) = 13 \times 0.7 + 3 \times 0.1 + 2 \times 0.2 = 9.8$, *and*
- $u(\gamma, w^*) = 10.5 \times 0.7 + 3 \times 0.1 + 2 \times 0.2 = 8.05$.

In the example above we are assuming that we know the decision-maker preference model. However, in the real world we don't know the weights vector representing the decision-maker preferences, but our interactive preference elicitation method can be used to estimate it, and to discover their optimal alternative.

2.2. Possibly optimal alternatives

We say that an alternative α is *optimal* in a set of alternatives A with respect to weights vector w if and only if $u(\alpha, w) \geq u(\beta, w)$ for any $\beta \in A$.

The following definition of *possibly optimal* alternatives is a common notion in multiple criteria decision-making and in works dealing with uncertain preferences [26, 27, 28, 29], and is related also to E-admissability [30, 31, 32].

Definition 1. *An alternative $\alpha \in A$ is possibly optimal in A, with respect to a given set Ω of weights vectors, if and only if there exists $w \in \Omega$ such that $u(\alpha, w) \geq u(\beta, w)$ for all $\beta \in A$, i.e., such that α is optimal in A with respect to w. We define $\mathrm{PO}(A, \Omega)$ as the set of all possibly optimal alternatives in A with respect to Ω.*

We can compute the set $\mathrm{PO}(A, \mathcal{W})$ of possibly optimal alternatives in A with respect to the space of user preferences $\mathcal{W}$ with a linear programming solver (see, e.g., [33]). Briefly, we can test if $\alpha \in \mathrm{PO}(A, \mathcal{W})$ evaluating the feasibility of the set of linear constraints $u(\alpha, w) \geq u(\beta, w)$ for all $\beta \in A \setminus \{\alpha\}$ with $w \in \mathcal{W}$. We focus only on the alternatives in $\mathrm{PO}(A, \mathcal{W})$ because the decision-maker's most preferred alternative must be optimal for the true preference w^*. Thus, we do not need to consider alternatives $\beta \notin \mathrm{PO}(A, \mathcal{W})$ and these alternatives can be filtered out as pre-processing.

2.3. Queries

Our approach focuses on binary queries, i.e., on asking the decision-maker to express their preferences with respect to pairwise comparisons of alternatives. We define a query as a pair (α, β) with $\alpha, \beta \in A$ (with $\alpha \neq \beta$), and the corresponding question for the decision-maker is '*Do you prefer α or β?*'. With a query (α, β) the decision-maker prefers α if and only if $w^* \cdot (\alpha - \beta) \geq 0$, and so otherwise, if $w^* \cdot (\alpha - \beta) < 0$ then the decision-maker prefers β. Many incremental preference elicitation procedures (see, e.g., [34, 35, 36, 37, 38])) iteratively ask this type of query with the purpose of reducing the space of feasible weights vectors by adding such hard constraints. However, a drawback of adding hard constraints to the set of feasible weights vectors is that we may exclude the optimal preference vector w^* if we receive an incorrect decision-maker answer.

2.4. User model

We assume a simple form of user model with two parameters: the preference vector $w^* \in \mathcal{W}$ and the noise parameter ρ with $0 \leq \rho < 1$, e.g., $\rho = 0.1$. Given a query (α, β), with probability $1-\rho$ the user will answer correctly, i.e., answer α if and only $w^* \cdot (\alpha - \beta) \geq 0$, and answer β otherwise; with probability ρ, the user will answer incorrectly, answering β iff $w^* \cdot (\alpha - \beta) \geq 0$. Note that neither w^* nor ρ are known by the learning system (only the answers to the queries).

Simplifying Assumption. We assume that, for each preference vector w, there is a unique element α_w in A that maximises $u(\alpha, w)$. With A consisting of random real-valued vectors this will almost certainly hold, and the assumption considerably simplifies the notation and the description of the algorithms, and improves the readability of the proofs from Section 6; for instance, it eliminates the possibility of a case when some query (α, β) keeps being asked, even though there is only one weights vector w still being considered, because $u(\alpha, w) = u(\beta, w)$. All the methods can be easily extended for situations in which this assumption does not hold, by changing the stopping criterion (Section 3.4) so that the main algorithm stops when there exists some alternative α_0 such that for every w in the set of most plausible preference models, α_0 maximises $u(\alpha, w)$.

3. The Idea Behind Our Approach

If it were the case that there exists a unique possibly optimal alternative α in A, i.e., if $\mathrm{PO}(A, \mathcal{W}) = \{\alpha\}$, then clearly we should recommend α, since it is optimal in A with respect to every preference vector w in $\mathcal{W}$ (and thus, in particular, with respect to the unknown user preference vector w^*). Similarly, if we were certain that the user was always answering our queries correctly, and $\mathcal{U}$ is the set of preference vectors compatible with the user's answers, and $\mathrm{PO}(A, \mathcal{U}) = \{\alpha\}$, then we should recommend α. In our context, where we are never certain about the correctness of the user's answers, we can adapt this idea by considering $\mathrm{PO}(A, \mathcal{U}')$, where $\mathcal{U}'$ is the set of most plausible preference vectors.

We are making no assumptions about a prior distribution over $\mathcal{W}$, so the plausibility of a preference vector w relates to how closely the user's answers are to those that would have been given if w were the true user preference vector w^* (and the user gave accurate answers). Thus, for a given query (α, β)

with answer α, we test if $w \cdot (\alpha - \beta) \geq 0$ to check if α is preferred to β according to the weights vector w. We suppose that the more inequalities are satisfied for a given $w \in \mathcal{W}$, the more plausible it is that w has a similar preference order to that of the true decision-maker preference. This is formalised with the function *mistakes*$(\cdot)$.

3.1. The function mistakes(w)

To find the preference vectors in $\mathcal{W}$ that correspond most closely to the decision-maker input preferences, we consider *mistakes*(w), which is defined to be the number of mistakes that the decision-maker would have made if $w \in \mathcal{W}$ were the true user preference vector w^*, i.e., the number of times the inequality $w \cdot (\alpha - \beta) \geq 0$ is not satisfied, for each query (α, β) (or (β, α)) with answer α. For example, with a query (α, β), if the decision-maker answers α, and $w \cdot (\alpha - \beta) < 0$, we increment *mistakes*(w) by one unit.

Because the user's answers can be incorrect, *mistakes*(w^*) will often be greater than zero. In particular, because we are considering a simple noisy user model, with a chance ρ of giving an incorrect answer independently for each query, the random variable *mistakes*(w^*) is binomially distributed with expected value ρK, where K is the number of queries asked.

Of course, we do not know *mistakes*(w^*) since the true user preference w^* is unknown; however, we can consider the set $\mathcal{W}^k_{min}$ of the most plausible preference vectors, including w such that *mistakes*(w) is within the threshold k of the minimal number of mistakes (defined below).

3.2. Finite approximation $\mathcal{W}'$ *of* $\mathcal{W}$

It is computationally convenient to approximate $\mathcal{W}$ by a finite set of points $\mathcal{W}'$. There are various ways this can be done; in our experiments we randomly sample elements of $\mathcal{W}$ using a uniform distribution[2] over the probability simplex $\mathcal{W}$; currently we use the same set $\mathcal{W}'$ throughout the whole iterative interaction process.

[2]Note that this is *not* equivalent to sampling each component $w(i)$ in the interval $[0, 1]$, and then normalising by dividing by $\sum_{i=1}^{p} w(i)$; the latter would pick elements w near $(1/p, \ldots, 1/p)$ with a higher probability density in comparison with elements near the edge of the region.

3.3. The set $\mathcal{W}^k_{min}$ of k-plausible preference vectors

Let μ be the minimum number of *mistakes*(w) over all $w \in \mathcal{W}'$:

$$\mu = \min_{w \in \mathcal{W}'} \textit{mistakes}(w).$$

We now define $\mathcal{W}^k_{min}$, the set of k-plausible preference vectors.

Definition 2. *For parameter $k \geq 0$ we define $\mathcal{W}^k_{min}$ to be all the preference points $w \in \mathcal{W}'$ such that mistakes$(w) \leq \mu + k$:*

$$\mathcal{W}^k_{min} = \{w \in \mathcal{W}' \mid \textit{mistakes}(w) \leq \mu + k\}.$$

We use $\mathcal{W}^k_{min}$ to generate queries for the decision-maker. Note that $\mathcal{W}^k_{min}$ depends on the randomly chosen set $\mathcal{W}'$, although our notation does not makes this explicit.

We say that w *and* w^* *differ on a query* (α, β) if either (a) $w \cdot \alpha \geq w \cdot \beta$ and $w^* \cdot \alpha < w^* \cdot \beta$; or (b) $w^* \cdot \alpha \geq w^* \cdot \beta$ and $w \cdot \alpha < w \cdot \beta$. The result below throws some light on how the set $\mathcal{W}^k_{min}$ will look after a sequence of queries.

Proposition 1. *Let $w \in \mathcal{W}$. Consider a sequence of queries, and let K_w be the number of queries in the sequence in which w and w^* differ. Let the random variable $\boldsymbol{X}^{w^*}_w$ be equal to mistakes(w) − mistakes(w^*). Then $\boldsymbol{X}^{w^*}_w \sim K_w - 2B(K_w, \rho)$, where $B(K_w, \rho)$ is a binomial distribution with K_w experiments and probability of success ρ. The expected value $E[\boldsymbol{X}^{w^*}_w]$ of $\boldsymbol{X}^{w^*}_w$ is equal to $K_w(1-2\rho)$, and the standard deviation of $\boldsymbol{X}^{w^*}_w$ equals $2\sqrt{K_w\rho(1-\rho)}$.*

Proof: Let us label the queries in the sequence on which w and w^* differ as (α_i, β_i) for $i = 1, \ldots, K_w$, and let the Boolean random variable $\mathbf{e}_i$ be such that $\mathbf{e}_i = 1$ if and only if the user answers query (α_i, β_i) incorrectly. The variables $\mathbf{e}_i$ for $i = 1, \ldots, K_w$ are independent with $\Pr(\mathbf{e}_i = 1) = \rho$. If $\mathbf{e}_i = 1$ then *mistakes*(w^*) is incremented and *mistakes*(w) is unchanged, so *mistakes*(w) − *mistakes*(w^*) is decremented; and if $\mathbf{e}_i = 0$ then *mistakes*(w) is incremented and so *mistakes*(w) − *mistakes*(w^*) is incremented. So, in both cases, $\mathbf{X}^{w^*}_w$ changes by $1 - 2\mathbf{e}_i$. (For the other queries, on which w and w^* do not differ, *mistakes*(w) − *mistakes*(w^*) is unchanged.) Thus, $\mathbf{X}^{w^*}_w$ = *mistakes*(w) − *mistakes*(w^*) is equal to $\sum_{i=1}^{K_w}(1-2\mathbf{e}_i)$, i.e., $K_w - 2\sum_{i=1}^{K_w}\mathbf{e}_i$. Therefore, $\mathbf{X}^{w^*}_w \sim K_w - 2B(K_w, \rho)$, because $\sum_{i=1}^{K_w}\mathbf{e}_i$ has the binomial distribution $B(K_w, \rho)$.

The expected value of $\sum_{i=1}^{K_w} \mathbf{e}_i$ is $K_w\rho$, and so $\mathrm{E}[\mathbf{X}_w^{w^*}] = K_w(1-2\rho)$. The variance of $\mathbf{e}_i$ is equal to $\mathrm{E}[(\mathbf{e}_i)^2] - (\mathrm{E}[\mathbf{e}_i])^2 = \rho - \rho^2$, and so the variance of $\mathbf{X}_w^{w^*}$, which equals the variance of $2\sum_{i=1}^{K_w} \mathbf{e}_i$, is $4K_w\rho(1-\rho)$; hence the standard deviation of $\mathbf{X}_w^{w^*}$ is equal to $2\sqrt{K_w\rho(1-\rho)}$. □

Using the notation of Proposition 1 we have the following consequence.

Corollary 1. *Assume $w^* \in \mathcal{W}'$.*

(i) If $\boldsymbol{X}_w^{w^*} > \kappa$ then $w \notin \mathcal{W}_{min}^{\kappa}$.

(ii) If $w \in \mathcal{W}_{min}^{\kappa}$ and $K_w(1-2\rho) > \kappa$ then the random variable $\boldsymbol{X}_w^{w^*}$ is at least $\frac{K_w(1-2\rho)-\kappa}{2\sqrt{K_w\rho(1-\rho)}}$ standard deviations less than its mean.

Proof: (i) Assume $\mathbf{X}_w^{w^*} > \kappa$. Since $w^* \in \mathcal{W}'$ we have $mistakes(w^*) \geq \mu$. Then $mistakes(w) - \mu \geq mistakes(w) - mistakes(w^*) = \mathbf{X}_w^{w^*} > \kappa$, so $mistakes(w) - \mu > \kappa$, and thus, $w \notin \mathcal{W}_{min}^{\kappa}$.
(ii) By part (i), $w \in \mathcal{W}_{min}^{\kappa}$ implies $\mathbf{X}_w^{w^*} \leq \kappa$. By Proposition 1, the random variable has mean $K_w(1-2\rho)$ and standard deviation of $2\sqrt{K_w\rho(1-\rho)}$, and $\frac{K_w(1-2\rho)-\mathbf{X}_w^{w^*}}{2\sqrt{K_w\rho(1-\rho)}} \geq \frac{K_w(1-2\rho)-\kappa}{2\sqrt{K_w\rho(1-\rho)}}$, so $\mathbf{X}_w^{w^*}$ is at least $\frac{K_w(1-2\rho)-\kappa}{2\sqrt{K_w\rho(1-\rho)}}$ standard deviations less than its mean. □

Corollary 1 implies that the w (in $\mathcal{W}'$) with high K_w will tend to not be in $\mathcal{W}_{min}^{\kappa}$. Indeed, if K_w is high then $K_w(1-2\rho)$, the expected value of $\mathbf{X}_w^{w^*}$, is high too. By Corollary 1(i), this means that w is not in $\mathcal{W}_{min}^{\kappa}$.

In particular, if w^* was in $\mathcal{W}'$ and κ was small (such as $\kappa \in \{0,1,2\}$), then it is very unlikely for w to be in $\mathcal{W}_{min}^{\kappa}$ unless K_w is fairly small. This is because if w were in $\mathcal{W}_{min}^{\kappa}$ then, by Corollary 1(ii), we would have the approximately normally distributed random variable $\mathbf{X}_w^{w^*}$ being at least $\frac{K_w(1-2\rho)-\kappa}{2\sqrt{K_w\rho(1-\rho)}}$ standard deviations less than its mean. For instance, with $\kappa = 2$ and $\rho = 0.1$ and $K_w = 10$ (respectively, $K_w = 15$), $\mathbf{X}_w^{w^*}$ will be more than 3 (respectively, 4.3) standard deviations from its mean.

The following result, which is quite similar in its proof to Proposition 1, is useful for the case when $w^* \notin \mathcal{W}'$.

Proposition 2. *Let $v, w \in \mathcal{W}$. Consider a sequence of queries, and, for $u \in \{v, w\}$ let K_u be the number of queries in the sequence in which u and w^* differ; let K_w^v be the number of queries in the sequence in which w differs from*

both w^* and v (and so v and w^* agree), and define K_v^w analogously. Then $K_w - K_v = K_w^v - K_v^w$. Let the random variable $\boldsymbol{Z}_w^v$ be equal to mistakes(w) − mistakes(v). Then $\boldsymbol{Z}_w^v = \boldsymbol{X}_w^{w^*} - \boldsymbol{X}_v^{w^*}$, and $\boldsymbol{Z}_w^v \sim (K_w - K_v) - 2B(K_w^v, \rho) + 2B(K_v^w, \rho)$, with the two binomial distributions being independent. The expected value $E[\boldsymbol{Z}_w^v]$ of $\boldsymbol{Z}_w^v$ is equal to $(K_w - K_v)(1 - 2\rho)$, and the standard deviation of $\boldsymbol{Z}_w^v$ equals $2\sqrt{(K_w^v + K_v^w)\rho(1-\rho)} \leq 2\sqrt{(K_w + K_v)\rho(1-\rho)}$.

Proof: K_w can be written as $K_w^v + L$, where L is the number of queries in which w and v both disagree with w^*. Similarly, $K_v = K_v^w + L$, so $K_w - K_v = K_w^v - K_v^w$ and $K_w \geq K_w^v \geq 0$ and $K_v \geq K_v^w \geq 0$.

Let us label the queries in the sequence on which w differs from both v and w^* as (α_i, β_i) for $i = 1, \ldots, K_w^v$, and let the Boolean random variable $\mathbf{e}_i$ be such that $\mathbf{e}_i = 1$ if and only if the user answers query (α_i, β_i) incorrectly.

Similarly, let us label the queries in the sequence on which v differs from both w and w^* as (α'_j, β'_j) for $j = 1, \ldots, K_v^w$, and let the Boolean random variable $\mathbf{e}'_j$ be such that $\mathbf{e}'_j = 1$ if and only if the user answers query (α'_j, β'_j) incorrectly.

If v differs from both w and w^* on a query then w agrees with w^*; hence, the variables $\mathbf{e}_i$ for $i = 1, \ldots, K_w^v$ and the variables $\mathbf{e}'_j$ for $j = 1, \ldots, K_v^w$, do not overlap, and so are all mutually independent, with $\Pr(\mathbf{e}_i = 1) = \Pr(\mathbf{e}'_j = 1) = \rho$. Also, $\mathbf{Z}_w^v$ is equal to $\sum_{i=1}^{K_w^v}(1 - 2\mathbf{e}_i) - \sum_{j=1}^{K_v^w}(1 - 2\mathbf{e}'_j) = K_w^v - K_v^w - 2\sum_{i=1}^{K_w^v}\mathbf{e}_i + 2\sum_{j=1}^{K_v^w}\mathbf{e}'_j$, i.e., $K_w - K_v - 2\sum_{i=1}^{K_w^v}\mathbf{e}_i + 2\sum_{j=1}^{K_v^w}\mathbf{e}'_j$. Therefore, $\mathbf{Z}_w^v \sim K_w - K_v - 2B(K_w^v, \rho) + 2B(K_v^w, \rho)$, involving independent binomial distributions, i.e., binomial distributions on independent variables.

The expected value of each $\mathbf{e}_i$ and $\mathbf{e}'_j$ are both ρ, and their variances are both $\rho(1-\rho)$. Hence, the expected value of $\mathbf{Z}_w^v$ is $K_w^v - 2K_w^v\rho - K_v^w + 2K_v^w\rho = (K_w^v - K_w^v)(1-2\rho) = (K_w - K_v)(1-2\rho)$. And the variance of $\mathbf{Z}_w^v$ is $4\rho(1-\rho)K_w^v + 4\rho(1-\rho)K_v^w = 4\rho(1-\rho)(K_w^v + K_v^w) \leq 4\rho(1-\rho)(K_w + K_v)$, and so the standard deviation of $\mathbf{Z}_w^v$ is at most $2\sqrt{(K_w + K_v)\rho(1-\rho)}$. □

Even if $w^* \notin \mathcal{W}'$, the expected value of $\mathbf{Z}_w^v = \mathbf{X}_w^{w^*} - \mathbf{X}_v^{w^*} =$ mistakes(w) − mistakes(v) is equal to $(K_w - K_v)(1 - 2\rho)$ (for $w, v \in \mathcal{W}'$), by Proposition 2; also, the standard deviation of $\mathbf{Z}_w^v$ increases only proportionally with $2\sqrt{K_w^v + K_v^w} \leq 2\sqrt{K_w + K_v}$. Random variable $\mathbf{Z}_w^v$ is approximately normal, as a difference between two independent distributions that are approximately normal. Thus, if K_w is substantially larger than K_v then $\mathbf{Z}_w^v$ will very likely be greater than κ, and so w will then not be in $\mathcal{W}_{min}^{\kappa}$. Thus, $\mathcal{W}_{min}^{\kappa}$ will tend to contain the $w \in \mathcal{W}'$ with smallest K_w, i.e., that differ on fewest queries with the true model w^*.

More generally, the weights vectors w that order the alternatives in A most similarly to how w^* orders them, will tend to be in $\mathcal{W}^{\kappa}_{min}$, increasingly so as we ask more queries. Therefore, the most plausible user models w, i.e., those most likely to be the true user preference model w^* (or close to it), are those with smaller values of *mistakes*(w), which is why $\mathcal{W}^{\kappa}_{min}$ may be considered as consisting of the most plausible user preference models.

3.4. Stopping criterion

Our algorithm makes use of $\mathcal{W}^{k}_{min}$, for $k = 0, \ldots, \kappa$ where we focus on $\kappa = 2$ in our experimental testing. The stopping criterion for our algorithm is that all the most plausible user preference models agree on which alternative α is best, i.e., $\mathrm{PO}(A, \mathcal{W}^{\kappa}_{min}) = \{\alpha\}$. As we show in Section 6, this can be made to hold eventually (with probability tending to one) with sufficient appropriately chosen queries. Generally our query strategies aim to ensure that the stopping criterion is satisfied as soon as possible. In particular, we can limit ourselves to queries of the form (α_v, α_w), where $v, w \in \mathcal{W}^{\kappa}_{min}$, since there always exists such a query if the stopping criterion is not yet satisfied, and such a query will with probability $1 - \rho$ increment the mistakes function for w, if w differs with w^* on this query (and thus, v agrees with w^*). If we were to keep repeating this query then, with high probability, w will be eliminated from $\mathcal{W}^{\kappa}_{min}$.

3.5. Structure of algorithm

Our active learning method is summarized by Algorithm 1. We select the query using one of the two methods described in Section 4. The decision-maker response will be used to update *mistakes*(w) for each $w \in \mathcal{W}'$ and to recompute $\mathcal{W}^{k}_{min}$ for $k \in \{0, 1, \ldots, \kappa\}$. The sets $\mathcal{W}^{k}_{min}$ will be part of the input for the query selection during the next step. We iterate this procedure until $\mathrm{PO}(A, \mathcal{W}^{\kappa}_{min})$ becomes a singleton set, say $\{\alpha\}$ (so also, $\mathrm{PO}(A, \mathcal{W}^{\kappa-1}_{min}) = \cdots = \mathrm{PO}(A, \mathcal{W}^{0}_{min}) = \{\alpha\}$); then α is our estimation of the most preferred alternative of the decision-maker, and we recommend it. Intuitively, this loop will tend to exclude weights vectors from $\mathcal{W}^{k}_{min}$ for $k \in \{0, 1, \ldots, \kappa\}$ whose preference orders are very different from the preferences of the decision-maker. Thus, weights vector in $\mathcal{W}^{k}_{min}$ for $k \in \{0, 1, \ldots, \kappa\}$ are more likely to have similar preference orders of that of w^*.

Algorithm 1

procedure RECOMMEND_ALTERNATIVE(A, $\mathcal{W}'$)
  repeat
    $q \leftarrow$ Select_query(A, $\mathcal{W}^k_{min}$ for $k \in \{0, 1, \ldots, \kappa\}$)
    Ask query q to the decision-maker
    Update $\mathcal{W}^k_{min}$ for $k \in \{0, \ldots, \kappa\}$
  until $|\mathrm{PO}(A, \mathcal{W}^\kappa_{min})| = 1$
  return The unique alternative in $\mathrm{PO}(A, \mathcal{W}^\kappa_{min})$

4. Query Selection

In this section we present our two methods for query selection. For $k = 0, \ldots, \kappa$, let $A^k = \mathrm{PO}(A, \mathcal{W}^k_{min})$. The definition of $\mathcal{W}^0_{min}$ implies that A^0 is not empty, and the definition of $\mathrm{PO}(A, \mathcal{W}^k_{min})$ and the fact that the sets $\mathcal{W}^k_{min}$ are nested then implies that $\emptyset \neq A^0 \subseteq A^1 \subseteq \cdots \subseteq A^\kappa \subseteq A$. Also, the stopping criterion occurs exactly when A^κ is a singleton, so, when generating a query, we always have $|A^\kappa| > 1$. In both query methods the query (α, β) is such that $\alpha \in A^0$ and $\beta \in A^k$ where k is minimal such that $|A^k| > 1$.

Method 1 (Max Preference Points). We select as a query a pair of alternatives (α, β) that are optimal in A with respect to a maximum number of $w \in \mathcal{W}^k_{min}$, focusing first on lower values of k. More precisely, for $k = 0, 1, \ldots, \kappa$, let $Opt^k(\alpha)$ be the number of preference points $w \in \mathcal{W}^k_{min}$ with α as the most preferred alternative α_w with respect to w, i.e., maximising $u(\alpha, w)$.

For clarity, in the description below we assume that $\kappa = 2$, but the method easily extends to $\kappa > 2$. We select the query (α, β) as follows (where one of these three conditions must hold, because the stopping criterion fails to hold):

- If $|\mathrm{PO}(A, \mathcal{W}^0_{min})| > 1$, select a query (α, β) with $\alpha, \beta \in \mathrm{PO}(A, \mathcal{W}^0_{min})$, $Opt^0(\alpha) \geq Opt^0(\gamma)$ and $Opt^0(\beta) \geq Opt^0(\gamma)$ for each $\gamma \in \mathrm{PO}(A, \mathcal{W}^0_{min}) \setminus \{\alpha, \beta\}$.
- If $\mathrm{PO}(A, \mathcal{W}^0_{min}) = \{\alpha_0\}$ and $|\mathrm{PO}(A, \mathcal{W}^1_{min})| > 1$, select a query (α_0, β) with $\beta \in \mathrm{PO}(A, \mathcal{W}^1_{min})$, $\beta \neq \alpha_0$ and $Opt^1(\beta) \geq Opt^1(\gamma)$ for each $\gamma \in \mathrm{PO}(A, \mathcal{W}^1_{min}) \setminus \{\alpha_0, \beta\}$.

- If $\mathrm{PO}(A, \mathcal{W}^0_{min}) = \mathrm{PO}(A, \mathcal{W}^1_{min}) = \{\alpha_0\}$ and $|\mathrm{PO}(A, \mathcal{W}^2_{min})| > 1$, select a query (α_0, β) with $\beta \in \mathrm{PO}(A, \mathcal{W}^2_{min})$, $\beta \neq \alpha_0$ and $Opt^2(\beta) \geq Opt^2(\gamma)$ for each $\gamma \in \mathrm{PO}(A, \mathcal{W}^2_{min}) \setminus \{\alpha_0, \beta\}$.

Method 2 (Best Split). In this case we select a query (α, β) with $\alpha, \beta \in \mathrm{PO}(A, \mathcal{W}^k_{min})$ that divides the preference points in $\mathcal{W}^k_{min}$ in two sets $\{w : (\alpha - \beta) \cdot w \geq 0\}$ and $\{w : (\alpha - \beta) \cdot w < 0\}$ with similar size. More precisely, first we compute a set of potential queries Q with the method described below.

- If $|\mathrm{PO}(A, \mathcal{W}^0_{min})| > 1$, set $k = 0$ and let $Q = \{(\alpha, \beta) : \alpha, \beta \in \mathrm{PO}(A, \mathcal{W}^0_{min}), \alpha \neq \beta\}$.
- Otherwise, write $\mathrm{PO}(A, \mathcal{W}^0_{min})$ as $\{\alpha_0\}$.
  - If $|\mathrm{PO}(A, \mathcal{W}^1_{min})| > 1$, set $k = 1$.
  - If $|\mathrm{PO}(A, \mathcal{W}^1_{min})| = 1$ set $k = 2$.
  - Set $Q = \{(\alpha_0, \beta) : \beta \in \mathrm{PO}(A, \mathcal{W}^k_{min}), \beta \neq \alpha_0\}$

The set Q contains all the candidate queries from which we need to select the next query to ask. Noting that each query divides, according to the two possible answers, the vectors $\mathcal{W}^k_{min}$ in two subsets, we aim at choosing the query that partitions $\mathcal{W}^k_{min}$ in the most even way. To do so, we select the query maximising the score computed with Algorithm 2 using as input Q and $\mathcal{W}^k_{min}$, with the value of k depending of which methods we used to generate Q. In Algorithm 2, for query $q = (\alpha, \beta)$ we define $\lambda_q = \alpha - \beta$.

Algorithm 2

```
1: procedure SELECT_QUERY(Q, W^k_min)
2:     query_score ← ∞
3:     for q in queries do
4:         min_score ← 0
5:         max_score ← 0
6:         for w in W^k_min do
7:             res = λ_q · w
8:             if res > 0 then
9:                 min_score ← min_score + 1
10:            if res < 0 then
11:                max_score ← max_score + 1
12:        maxmin_score ← max(min_score, max_score)
13:        if maxmin_score < query_score then
14:            query ← q
15:            query_score ← maxmin_score
       return query
```

If there is a tie in the score then the tie can be broken arbitrarily. In the experimental testing we use the first query found that achieves the maximum score, and similarly for Method 1.

With both Methods 1 and 2, if $|\mathrm{PO}(A, \mathcal{W}^0_{min})| > 1$ we will choose a query (α_v, α_w) with $v, w \in \mathcal{W}^0_{min}$, which will then reduce $\mathcal{W}^0_{min}$ and thus $|\mathrm{PO}(A, \mathcal{W}^0_{min})|$ in the next stage. $\mathcal{W}^0_{min}$ will continue to reduce until $\mathrm{PO}(A, \mathcal{W}^0_{min})$ equals a singleton $\{\alpha_0\}$. At this point the algorithm will ask a query of the form (α_0, β) for some $\beta \in \mathrm{PO}(A, \mathcal{W}^\kappa_{min})$. However, if the user responds that β is preferred to α_0 then at the next stage we may have $|\mathrm{PO}(A, \mathcal{W}^0_{min})| > 1$ again, and, over the following queries, $\mathrm{PO}(A, \mathcal{W}^0_{min})$ will again reduce to a singleton, and so on.

Issues caused by the finite approximation $\mathcal{W}'$ of $\mathcal{W}$

The first issue is that our approach can only recommend an alternative that is optimal with respect to a preference vector in the finite set $\mathcal{W}'$ i.e., an element of $\mathrm{PO}(A, \mathcal{W}')$. Hence, if it turns out that the user's optimal alternative, i.e., α^* $(= \alpha_{w^*})$, is not in $\mathrm{PO}(A, \mathcal{W}')$, then the algorithms will not recommend the correct answer, α^*.

A second issue is that we may have α^* in $\mathrm{PO}(A, \mathcal{W}')$ with $\alpha^* = \alpha_v$ for some $v \in \mathcal{W}'$, but where the preference ordering corresponding to v

has significant differences with the user's preference ordering, even though both rate α^* optimally in A. For example, if $A = \{\alpha = (1, 0, 0), \beta = (0.5, 0.3, 0.2), \gamma = (0.5, 0.2, 0.3)\}$, $w^* = (0.5, 0.5, 0)$ and $v = (0.5, 0, 0.5)$, we have $\alpha = \arg\max_{\delta \in A} w^* \cdot \delta = \alpha^* = \alpha_v = \arg\max_{\delta \in A} v \cdot \delta$, but $w^* \cdot (\beta - \gamma) \geq 0$ and $v \cdot (\beta - \gamma) < 0$. Thus, if the decision-maker answers the query (β, γ) correctly (with respect to w^*), *mistakes*(v) would increase by one unit. This type of situation may lead to *mistakes*(v) being significantly greater than *mistakes*(w^*), and even than *mistakes*(w) for some $w \in \mathcal{W}'$ with $\alpha_w \neq \alpha^*$. This would then mean that $v \notin \mathcal{W}^2_{min}$; if $\mathcal{W}'$ does not contain w^* or similar elements then we might well then have $\mathrm{PO}(A, \mathcal{W}^2_{min}) = \{\alpha_w\} \not\ni \alpha^*$, and so the system would recommend an incorrect alternative. However, this phenomenon does not happen very often, especially when $\mathcal{W}'$ is larger, because then there will tend to be a preference point in $\mathcal{W}'$ fairly close to w^*, and with a similar preference ordering.

Despite these issues, we will see in Section 5 that our algorithms still manage to have high accuracy. Our approach was tested 1,800 times, and returned the actual most preferred alternative of the decision-maker 1,774 times (1,292 times out of 1,300 when the user noise was 0.1).

The precise distribution of points in $\mathcal{W}'$ is not too important, but it is important that there are not big gaps between the elements of $\mathcal{W}'$ (as would be the case, for instance, if we set $\mathcal{W}'$ to be the set of extreme points of $\mathcal{W}$), since if w^* were far from any element of $\mathcal{W}'$, the user's optimal alternative may very well not even be in $\mathrm{PO}(A, \mathcal{W}')$, which would mean that the algorithm would not return the user's optimal alternative α^*.

Large distances between the elements of $\mathcal{W}'$ should still be avoided even if α^* is guaranteed to be in $\mathrm{PO}(A, \mathcal{W}')$. Consider for example a set $\mathcal{W}'$ that is built by picking d representatives (with $d \geq 1$) from each optimality polyhedron, ensuring $\mathrm{PO}(A, \mathcal{W}') = A$. Because the optimality polyhedrons can have vastly varying sizes, it is possible for w^* to be far away from the nearest element of $\mathcal{W}'$, and to not have any element of $\mathcal{W}'$ with a similar preference ordering (and therefore similar answers to queries) as w^*, making it more likely that the second issue discussed above will lead to a very poor alternative being recommended to the user. In contrast, our construction of $\mathcal{W}'$ uniformly samples over $\mathcal{W}$, so we can expect some elements of $\mathcal{W}'$ to behave similarly as w^*, minimising the impact caused by the second issue above.

In theory one might consider applying a version of the approach in this paper with $\mathcal{W}'$ being the whole of $\mathcal{W}$ rather than a finite subset of $\mathcal{W}$.

However, there will be serious computational difficulties in this case, arising, in particular, from the fact that the set $\mathcal{W}^0_{min}$, of all $w \in \mathcal{W}$ minimising *mistakes*(w), will typically be far from convex. For arbitrary subset S of the queries define the convex set $\mathcal{W}(S)$ to be all $w \in \mathcal{W}$ such that for every query in S, w would answer differently from the received answer, and for all queries not in S, w would give the same answer as the received answer. Hence *mistakes*$(w) = |S|$ for all $w \in \mathcal{W}(S)$. We then have that $\mathcal{W}^0_{min}$ is equal to the union of $\mathcal{W}(S)$ over all subsets S of cardinality μ, and so is the union of a potentially very large number of convex sets; this will tend to make the computation of e.g., $\mathrm{PO}(A, \mathcal{W}^0_{min})$ very costly when the minimum number μ of mistakes is not very small.

5. Experimental Results

In this section we discuss the results of the experimental testing of our approach applied to randomly generated decision problems.

A random problem is represented by a random set A of possibly optimal utility vectors[3]and a simulated decision-maker with utility function $u(\alpha, w^*)$. The goal is to find the most preferred alternative of the decision-maker, i.e., the alternative $\alpha_{w^*} \in A$ maximising $u(\alpha, w^*)$ for any $\alpha \in A$. We simulate a decision-maker for each experiment generating a random weights vector w^*. With a noise-free user model, the simulated decision-maker response to a comparison query of two alternatives (α, β) will be α if $w^* \cdot \alpha \geq w^* \cdot \beta$, and β otherwise. However, we want to simulate noisy user responses, therefore we take into account a fixed probability ρ (e.g., $\rho = 0.1$) of receiving the incorrect answer.

Let $\mathrm{PO}(A, \mathcal{W})$ be the set of possibly optimal alternatives in A with respect to $\mathcal{W}$. We generate random sets of n alternatives starting with $A = \emptyset$ as follows.

1. Generate a random vector $\alpha = (\alpha_1, \ldots, \alpha_p)$ with $\alpha_i \geq 0$ and $\sum \alpha_i = 1$.
2. $A = A \cup \{\alpha\}$ if $\mathrm{PO}(A \cup \{\alpha\}, \mathcal{W}) = A \cup \{\alpha\}$.
3. Repeat until $|A| = n$.

[3]Only the possibly optimal alternatives in a set A of alternatives are relevant, so if we didn't enforce that all alternatives are possibly optimal, then we would effectively be dealing with a (perhaps very much) smaller problem.

Given a random user with weights vector w, the most preferred alternative is $\alpha_w = \arg\max_{\alpha \in A} u(\alpha, w)$. As discussed at the end of Section 4, since we consider only a finite subset $\mathcal{W}' \subseteq \mathcal{W}$ of weights vectors, we may not be able to recommend α^* $(= \alpha_{w^*})$ since there may not be $w \in \mathcal{W}'$ such that $\alpha^* = \arg\max_{\alpha \in A} u(\alpha, w)$.

In Table 1 we show the average number of queries, the average iteration time and the accuracy. The accuracy is the fraction of experiments in which the correct alternative was recommended, i.e., the optimal alternative α_{w^*} in A according to the unknown true user model w^*. The results are an average of 100 experiments with random sets $\mathcal{W}'$ of 4000 weights vectors, and input sets A of 1000 random possibly optimal alternatives. We always used the same set A of alternatives, and 100 random user models w^*. The accuracy was high and we always had at least one $w \in \mathcal{W}'$ with $\alpha_w = \alpha_{w^*}$.

Table 1: Experimental results w.r.t. the number of criteria p, for Method 1 with $|A| = 1000$, $\rho = 0.1$ and $|\mathcal{W}'| = 4000$.

p	Queries	Time[s]	Accuracy
3	10.66	0.038	1.00
4	22.16	0.038	1.00
5	30.44	0.038	0.97
6	35.88	0.038	1.00

We have similar results for Method 2 in Table 2; this requires on average a smaller number of queries to converge, but is much slower than the first method.

In general, as the number of criteria increases, so does the number of queries.

Regarding the iteration time, Method 1 seems to be roughly independent of the number of criteria. This is because the most time-consuming operation in this case is the update of *mistakes*(w) for each $w \in \mathcal{W}$, which is not affected much by the number of criteria since the most expensive operation is the computation of $|\mathcal{W}'|$ dot products. For example, with $p = 5$, we required on average 0.34ms to compute the query, 36.2ms to update *mistakes*(w) for all $w \in \mathcal{W}$ and 1.05ms to compute the three sets $\mathcal{W}^k_{min}$, $k = 0, 1, 2$.

On the other hand, most of the iteration time for Method 2 is taken by the computation of the queries since the number of queries we evaluate

depends on the size of the sets $\mathcal{W}^k_{min}$, which increases with respect to the number of criteria. For example, with $p = 5$, we required on average 2.53s to compute the query, 35.6ms to update $mistakes(w)$ for each $w \in \mathcal{W}$ and 1ms to compute the sets $\mathcal{W}^k_{min}$.

Table 2: Experimental results for Method 2; $|A| = 1000$, $\rho = 0.1$ and $|\mathcal{W}'| = 4000$.

p	Queries	Time[s]	Accuracy
3	11.55	0.289	1.00
4	17.55	0.508	0.99
5	25.84	2.574	1.00
6	28.42	4.789	1.00

Table 3: Experimental results for regret-based elicitation with the current solution query strategy; $|A| = 1000$ and $\rho = 0.1$.

p	Queries	Time[s]	Accuracy
3	4.58	4.12	0.60
4	7.82	3.41	0.34
5	11.93	3.66	0.34
6	16.69	3.69	0.28

We provide, for comparison, the simulation results obtained by using state-of-the-art elicitation methods on the same datasets. In Table 3 we show the performance of interactive elicitation based on minimax regret with queries generated using the current solution strategies. Unsurprisingly, regret-based elicitation provides low accuracy as it cannot appropriately deal with user noise.

Table 4: Experimental results for Bayesian elicitation with the queries generated with greedy maximisation of value of information; $|A| = 1000$ and $\rho = 0.1$.

p	Queries	Time[s]	Accuracy
3	12.05	2.62	0.97
4	17.93	3.68	0.99
5	26.46	5.07	0.97
6	35.86	5.50	1.00

In Table 4 we consider the Bayesian elicitation method from [14]; queries are chosen to maximise Expected Value of Selection (EUS), a proxy of myopic value of information, using greedy maximisation; Bayesian updates are performed using Monte Carlo methods with 50000 particles. The elicitation stops when the expected loss is less than 0.001; when this happens the alternative with highest expected utility is recommended. As the table shows, the Bayesian method achieves high accuracy but at the cost of large computation times (higher accuracy may be obtained using more particles, but computation time will increase even further).

We now consider again our own approach, focusing on Method 1 for generating queries.

In Table 5 we show the performances of our method with respect to the size of $\mathcal{W}'$. The accuracy increases with increasing $|\mathcal{W}'|$, as does the computation time, and, to a lesser extent, the average number of queries.

Table 5: Experimental results w.r.t. the number $|\mathcal{W}'|$ of user preference models, for Method 1 with $|A| = 1000$, $\rho = 0.1$ and $p = 5$.

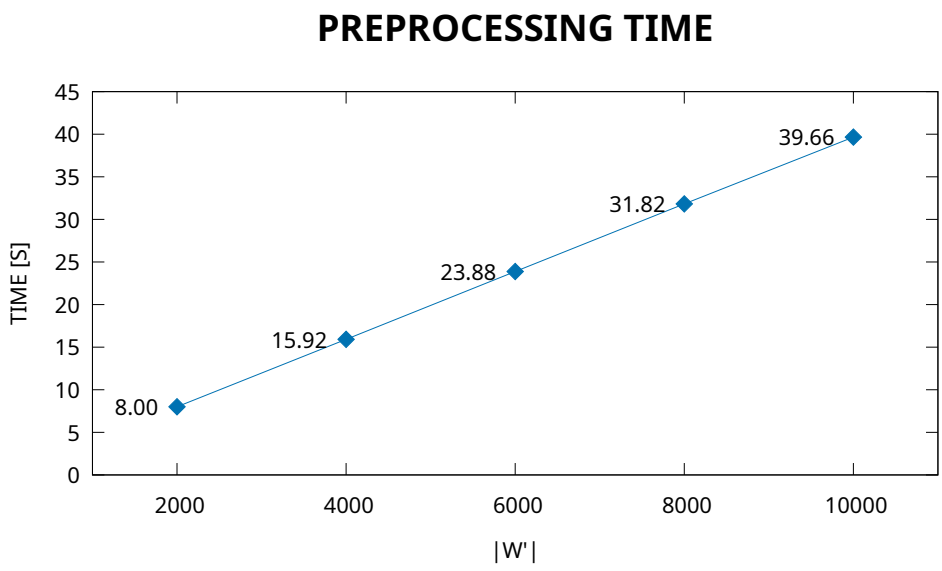
$\lvert\mathcal{W}'\rvert$	Queries	Time[s]	Accuracy
2000	27.28	0.019	0.97
4000	30.44	0.038	0.97
6000	32.71	0.057	1.00
8000	33.12	0.075	0.99
10000	34.50	0.096	1.00

We also tested the performances with respect to the number of alterna-

tives $|A| \in \{200, 400, 600, 800, 1000\}$ with $|\mathcal{W}'| = 4000$, $p = 4$ and $\rho = 0.1$. However, we didn't notice any significant difference in terms of accuracy and execution time. The average number of queries was affected slightly more, i.e., between 20.67 and 22.16, with lower values for lower $|A|$.

For an efficient implementation of Method 1, it is convenient to find the best alternative α_w for each $w \in \mathcal{W}'$ since this will speed up the computation of $\text{Opt}^k(\alpha)$ at each iteration. Thus, Method 1 requires this initial preprocessing which has a complexity proportional to $|A| * |\mathcal{W}'|$. Note that this operation needs to be performed only once for each pair A and $\mathcal{W}'$. Thus we can do it offline and re-use the results for any decision-maker. In Figure 1 we show the average preprocessing time (in seconds) with respect to the cardinality $|\mathcal{W}'|$ of the finite set $\mathcal{W}'$, based on 100 experiments with $|A| = 1000$ and $p = 5$. The number of criteria do not greatly affect this operation, which was confirmed by our experiments with $p \in \{3, 4, 5, 6\}$ where we obtained on average a maximum difference of 0.5s.

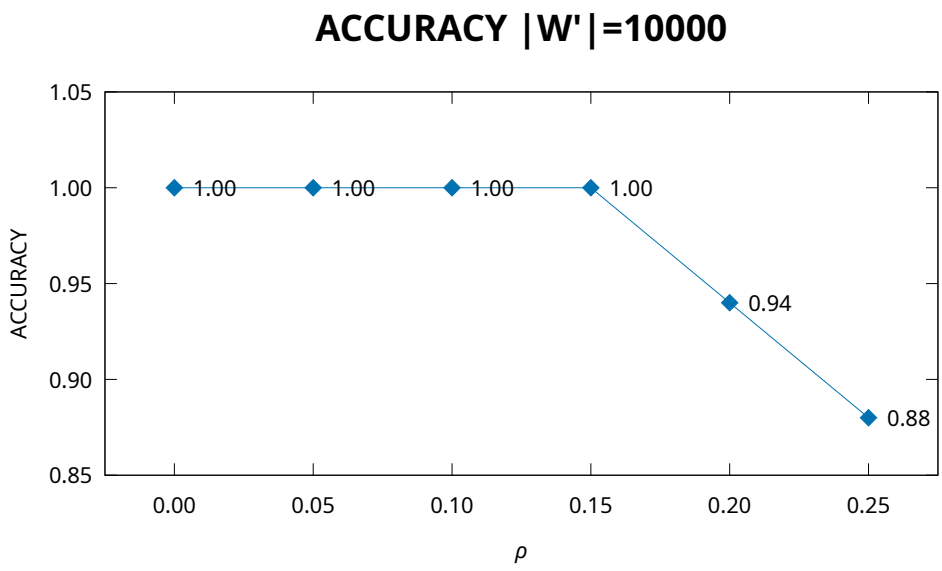
Figure 1: Average preprocessing time for Method 1 over 100 experiments with $|A| = 1000$ and $p = 5$.



In Figure 2 we show the accuracy varying with respect to the user noise ρ, with $|A| = 1000$, $|\mathcal{W}'| = 10000$ and $p = 4$. Unsurprisingly, the accuracy decreases with increasing user noise ρ. However, this picture shows that our

model can achieve good performance also with more noisy responses. In this case, the average query time was 0.093s. This dropping off of the accuracy for larger ρ tallies with our analysis around Proposition 1, and, to maintain very high accuracy, we will need to increase the parameter κ in our algorithm (from its current value of 2), in order to make the stopping condition harder to satisfy.

Figure 2: Accuracy varying with user noise ρ over 100 experiments with $|A| = 1000$, $|\mathcal{W}'| = 10000$ and $p = 4$.



6. Analysis of Algorithmic Approach

In this section we derive theoretical results about the convergence of our algorithms in Sections 3 and 4, and those of a similar form to our own.

6.1. Preliminaries and assumptions

Our algorithms generate a sequence of queries which the user model answers. Our analysis applies to more general ways of generating the queries, in fact to arbitrary ways of generating queries, with the exception that Theorems 2 and 3 assume that if the algorithm never terminates then each possible query is asked an infinite number of times.

At each point in the running of the algorithm there is a sequence of queries $q_j = (\alpha_j, \beta_j)$ for $j = 1, \ldots, r$, and the corresponding answers $\gamma_j \in \{\alpha_j, \beta_j\}$. The correct answer to query q_j is α_j if $w^* \cdot \alpha_j \geq w^* \cdot \beta_j$. For simplicity we assume that for each w in the finite set $\mathcal{W}'$ and each pair α, β of different alternatives, we have $w \cdot \alpha \neq w \cdot \beta$.

The stopping condition is that $\mathrm{PO}(A, \mathcal{W}'^{\kappa}_{min})$ is a singleton set $\{\alpha\}$, where $\mathcal{W}'^{\kappa}_{min}$ is the set of all $w \in \mathcal{W}'$ such that *mistakes*(w) are within κ of the minimum value $\mu = \min_{w' \in \mathcal{W}'}$ *mistakes*(w').

Assumption on finite set $\mathcal{W}'$. In order to obtain our results, we make the (strong) assumption that $\mathcal{W}'$ contains at least one element that is equivalent to w^*, i.e., gives the same answers to queries as w^* gives. Of course, in practice this cannot be guaranteed; however, we will typically have an element in $\mathcal{W}'$ that is close to w^*, and will tend to behave similarly.

For each $j = 1, \ldots, r$ the Boolean random variable $\mathbf{e}_j$ represents whether or not the user model answers correctly, with $\mathbf{e}_j = 0$ meaning correctly, and $\mathbf{e}_j = 1$ meaning that the user model answers incorrectly.

Assumptions on user model. We assume that the random variables $\mathbf{e}_j$ are mutually independent, and with $\Pr[\mathbf{e}_j = 1] = p_j$, where for all j, p_j is bounded above by some value $\rho < 0.5$.

In our algorithms in Sections 3 and 4 we in fact have $p_j = \rho$ for all j. But our analysis in this section applies to arbitrary error probabilities $p_j \leq \rho$ for the user model, which may depend on the particular query (or other factors). For example, one could model the error probability p_q for a query q by $p_q = 0.08 - \frac{0.08}{2^{1+err_q}}$, with err_q being the number of times that the decision-maker has incorrectly answered query q until now. This example model takes into account the fact that a query is usually more likely to be given an incorrect answer when the decision-maker has already made an error on it. In this model the random variables $\mathbf{e}_j$ are not mutually independent, but the model is still covered by our results because all probabilities $\Pr[\mathbf{e}_j = 1]$ have an upper bound of $\rho = 0.08 < 0.5$.

The functions *mistakes*(w) for $w \in \mathcal{W}'$ are random variables, depending on the variables $\mathbf{e}_j$. In particular, *mistakes*(w^*) is equal to $\sum_{j=1}^{r} \mathbf{e}_j$.

A key random variable for this form of algorithm is $\mathbf{X}_w^{w^*}$ equalling *mistakes*$(w)-$ *mistakes*(w^*). If this becomes greater than κ for each $w \in \mathcal{W}'$ not equivalent with w^*, and $\mathcal{W}'$ contains some element w' equivalent with w^*, then $\mathcal{W}'^{\kappa}_{min}$

will only contain elements equivalent with w^*, which then implies our stopping criterion $\mathrm{PO}(A, \mathcal{W}^{\kappa}_{min}) = \{\alpha^*\}$, and so the algorithm outputs the correct optimal alternative α^*.

Let $S^{w^*}_w$ be the set of $j \in \{1, \ldots, r\}$ such that w differs from w^* on the associated query $q_j = (\alpha_j, \beta_j)$. Let $K_w = |S^{w^*}_w|$ be the number of such components. Let $\mathbf{Y}^{w^*}_w$ equal $\sum_{j \in S^{w^*}_w} \mathbf{e}_j$, which is the number of incorrect user answers among components j in $S^{w^*}_w$.

For $j \in S^{w^*}_w$: if $\mathbf{e}_j = 0$ this increments *mistakes*(w) and doesn't change *mistakes*(w^*); if $\mathbf{e}_j = 1$ this increments *mistakes*(w^*) and doesn't change *mistakes*(w). For the particular case where $\mathbf{e}_j = 0$ for all $j \in S^{w^*}_w$, i.e., all the user answers are correct, then *mistakes*$(w^*) = 0$ and *mistakes*$(w) = K_w$ and thus, $\mathbf{X}^{w^*}_w = K_w$. For each $j \in S^{w^*}_w$, changing $\mathbf{e}_j = 0$ to $\mathbf{e}_j = 1$ increments *mistakes*(w^*) and decrements *mistakes*(w). Thus, as in the proof of Proposition 1, we have

$$\mathbf{X}^{w^*}_w = K_w - 2\mathbf{Y}^{w^*}_w.$$

Because the random variables $\mathbf{e}_j$ are mutually independent, $\mathbf{Y}^{w^*}_w$ is a sum of independent Bernouilli trials, and so has what is known as a Poisson Binomial distribution. If $p_j = \rho$ for all j then $\mathbf{Y}^{w^*}_w$ has a binomial distribution $B(K_w, \rho)$ with K_w experiments and probability of success[4] ρ. Then, $\Pr[\mathbf{X}^{w^*}_w \leq x]$ is equal to $\Pr[K_w - 2\mathbf{Y}^{w^*}_w \leq x] = \Pr[\mathbf{Y}^{w^*}_w \geq \frac{1}{2}(K_w - x)]$.

For the general case of $p_j \leq \rho$ we have for all y, $\Pr[\mathbf{Y}^{w^*}_w \geq y] \leq \Pr[\mathbf{Y} \geq y]$, where $\mathbf{Y} \sim B(K_w, \rho)$ and thus, $\Pr[\mathbf{X}^{w^*}_w \leq x] \leq \Pr[K_w - 2\mathbf{Y} \leq x]$.

We have thus shown the following result.

Lemma 1. *For any integer x:*

(i) If $\Pr[\boldsymbol{e}_j] = \rho$ for all j (i.e., a fixed probability ρ for making a mistake at each query) then $\boldsymbol{Y}^{w^*}_w$ has a binomial distribution: $\boldsymbol{Y}^{w^*}_w \sim B(K_w, \rho)$ and $\Pr[\boldsymbol{X}^{w^*}_w \leq x] = \Pr[K_w - 2\,\boldsymbol{Y}^{w^*}_w \leq x]$.

(ii) If $\Pr[\boldsymbol{e}_j] = p_j \leq \rho < \frac{1}{2}$ then $\Pr[\boldsymbol{X}^{w^*}_w \leq x] \leq \Pr[K_w - 2\,\boldsymbol{Y} \leq x]$, where $\boldsymbol{Y} \sim B(K_w, \rho)$.

The next result, based on a bound on the tail of a binomial distribution, is used to bound the distribution of the random variable $\mathbf{X}^{w^*}_w$.

[4] *Success* for the associated Bernouilli trials corresponds to the user model giving an incorrect response.

Lemma 2. *Consider any integers n, m, x with $0 \le n \le m$ and $-m < x < n(1-2\rho)$. Let random variable $\boldsymbol{Y}$ have a binomial distribution $B(m,\rho)$ with m experiments and probability of success ρ. We have* $\Pr[m - 2\boldsymbol{Y} \le x] \le \exp\bigl(-2n(\frac{1}{2} - \rho - \frac{x}{2n})^2\bigr)$.

Proof: Hoeffding's theorem (Theorem 1 of [39]) implies that for all $0 < t < 1-\rho$, we have $\Pr[\frac{1}{m}\mathbf{Y} - \rho \ge t] \le \exp(-2mt^2)$, and so

$$\Pr[\mathbf{Y} \ge m(t+\rho)] \le \exp(-2mt^2).$$

Now, $m - 2\mathbf{Y} \le x$ if and only if $\mathbf{Y} \ge \frac{1}{2}(m-x)$, and so

$$\Pr[m - 2\mathbf{Y} \le x] = \Pr[\mathbf{Y} \ge \frac{1}{2}(m-x)].$$

If we set $t = \frac{1}{2} - \rho - \frac{x}{2m}$ then $m(t+\rho) = \frac{1}{2}(m-x)$, and so, by Hoeffding's theorem,

$$\Pr[\mathbf{Y} \ge \frac{1}{2}(m-x)] \le \exp(-2mt^2) = \exp\left(-2m\left(\frac{1}{2} - \rho - \frac{x}{2m}\right)^2\right),$$

as long as $0 < t < 1-\rho$ i.e., $-m < x < m(1-2\rho)$; the latter holds because of the assumption that $-m < x < n(1-2\rho)$ and because $n(1-2\rho) \le m(1-2\rho)$.

Since $n \le m$ we also have

$$\exp\left(-2m\left(\frac{1}{2} - \rho - \frac{x}{2m}\right)^2\right) \le \exp\left(-2n\left(\frac{1}{2} - \rho - \frac{x}{2n}\right)^2\right),$$

because decreasing m decreases both $2m$ and $(\frac{1}{2} - \rho - \frac{x}{2m})^2$, with the assumption $x < n(1-2\rho)$ ensuring that $\frac{1}{2} - \rho - \frac{x}{2n} > 0$. Combining the last three displayed equations gives the result. □

In the proposition below, part (i) bounds the probability of the event $\mathbf{X}_w^{w^*} < -\kappa$ for $w \in \mathcal{W}'$; this event implies that any w' equivalent with w^* would not be in the set $\mathcal{W}_{min}^{\kappa}$ of best models, so we want its probability to be small. Part (ii) bounds the probability of the event $\mathbf{X}_w^{w^*} \le \kappa$; if the algorithm fails to terminate then this event must hold for some $w \in \mathcal{W}'$ at each point of the algorithm.

Proposition 3. *Let $m = K_w$, and n be such that $\frac{\kappa}{1-2\rho} < n \le m$.*

(i) $\Pr[\boldsymbol{X}_w^{w^*} < -\kappa] \leq \lambda^m$ *where* $\lambda = \exp(-2(\frac{1}{2} - \rho)^2) < 1$.

(ii) $\Pr[\boldsymbol{X}_w^{w^*} \leq \kappa] \leq \exp\bigl(-2n(\frac{1}{2} - \rho - \frac{\kappa}{2n})^2\bigr)$.

Proof: (i): Since $\kappa > 0$ we have $\Pr[\mathbf{X}_w^{w^*} < -\kappa] \leq \Pr[\mathbf{X}_w^{w^*} \leq 0]$. We have $-m < 0 < m(1-2\rho)$, because $\rho < \frac{1}{2}$, and so Lemma 2 can be applied with $x = 0$. Combining Lemma 1(ii) and Lemma 2 using $x = 0$ and $m = n$ gives $\Pr[\mathbf{X}_w^{w^*} \leq 0] \leq \exp\bigl(-2m(\frac{1}{2} - \rho)^2\bigr) = \lambda^m$.

(ii): This follows immediately from Lemma 1(ii) and Lemma 2 using $x = \kappa$, because $-m < \kappa < n(1-2\rho)$, which ensures that Lemma 2 can be applied with $x = \kappa$. □

6.2. Showing that the probability of returning an incorrect alternative tends to zero as $\kappa \to \infty$

For a given κ, let P_κ be the probability that the algorithm returns the correct alternative α^*, let P'_κ be the probability that the algorithm returns an alternative different from α^*, and let P^∞_κ be the probability that the algorithm never terminates. Because the algorithm only terminates when returning an alternative, we have $P_\kappa + P'_\kappa + P^\infty_\kappa = 1$. We would like P_κ to be as close to 1 as possible. In Subsection 6.3, we will show that $P^\infty_\kappa = 0$ for every κ. In the current subsection we show that we can select κ such that P'_κ is arbitrarily small.

There is always a possibility that we are very unlucky with the random incorrect responses from the user model: that a user model $w' \in \mathcal{W}'$ equivalent with w^* is removed from $\mathcal{W}^\kappa_{min}$, and that $\mathcal{W}^\kappa_{min}$ only contains user models w for which some different alternative β is optimal; then the algorithm will return β. However, this kind of scenario is very unlikely, increasingly so as κ increases. We show an asymptotic result here.

Theorem 1. *Given the assumptions in Section 6.1, let* P'_κ *be the probability that the algorithm returns an alternative different from* α^*. *Then* $\lim_{\kappa\to\infty} P'_\kappa = 0$.

Proof: For $w \in \mathcal{W}'$ let $E(w)$ be the event that at some point in the running of the algorithm, $\mathit{mistakes}(w^*) - \mathit{mistakes}(w) > \kappa$. Because $\mathcal{W}'$ is assumed to contain an element equivalent with w^*, this must happen for some $w \in \mathcal{W}'$ for the algorithm to return an alternative different from α^*. Then $P'_\kappa \leq \Pr[\bigcup_{w\in\mathcal{W}'} E(w)] \leq \sum_{w\in\mathcal{W}'} \Pr[E(w)]$. We will give a function $f(\kappa)$, tending

to zero as κ tends to infinity, such that for all $w \in \mathcal{W}'$, $\Pr[E(w)] \leq f(\kappa)$, which will show that $P'_\kappa \leq |\mathcal{W}'| f(\kappa)$, and so, $\lim_{\kappa\to\infty} P'_\kappa = 0$.

For $m = 1, 2, \ldots$, and $w \in \mathcal{W}'$, let $E_m(w)$ be the event that *mistakes*(w^*) − *mistakes*$(w) > \kappa$ after m queries on which w and w^* differ, i.e., that $\mathbf{X}_w^{w^*} < -\kappa$ when $K_w = m$. Hence, $E(w) = \bigvee_{m=1}^{\infty} E_m(w)$. Because $E_m(w)$ is impossible unless $m > \kappa$ we have $\Pr[E(w)] \leq \sum_{m>\kappa} \Pr[E_m(w)]$.

By Proposition 3(i), $\Pr[E_m(w)] = \Pr[\mathbf{X}_w^{w^*} < -\kappa] \leq \lambda^m$ where $\lambda = \exp(-2(\frac{1}{2} - \rho)^2)) < 1$, and so $\Pr[E(w)] \leq \sum_{m\geq\kappa+1} \Pr[E_m(w)] \leq \frac{\lambda^{\kappa+1}}{1-\lambda}$. Therefore, $P'_\kappa \leq |\mathcal{W}'| \frac{\lambda^{\kappa+1}}{1-\lambda}$, which tends to zero as κ tends to infinity. □

6.3. About the probability that the algorithm will terminate

We assume that if the total number of queries generated is infinite (i.e., if the stopping condition never holds), then each individual query is asked an infinite number of times. This is not difficult to enforce, for example the algorithm could periodically ask the query that has been asked the fewest times so far.

Theorem 2. *Along with the assumptions in Section 6.1, assume that for every sequence SEQ of queries asked by the algorithm, either SEQ is finite, or SEQ contains each possible query an infinite number of times. Then the probability that the algorithm terminates after at most q queries tends to 1 as q tends to infinity.*

Proof: For $n = 1, 2, \ldots$ let Stage n refer to a point in which the current sequence includes at least n repeats of each query, with the most recent query having exactly n repeats; this can only occur at most once for each n in the running of the algorithm, because starting from the next query, the most recent query will always have at least $n + 1$ repeats.

If the stopping criterion never holds then an infinite number of queries are asked and, because of our assumption, each individual query is asked an infinite number of times. So if the algorithm never stops, then for each n, Stage n occurs at some point. Let G_n be the event that the algorithm reaches Stage n, and at that point the stopping criterion does not hold.

For each $w \in \mathcal{W}'$ let $G_n(w)$ be the event that G_n holds and that at Stage n, *mistakes*(w) − *mistakes*$(w^*) \leq \kappa$, i.e., *mistakes*(w) − *mistakes*$(w') \leq \kappa$ for any $w' \in \mathcal{W}'$ equivalent with w^*. So, if G_n holds but $G_n(w)$ fails to hold then $w \notin \mathcal{W}^\kappa_{min}$, since we are assuming that there exists some $w' \in \mathcal{W}'$ that is

equivalent with w^*. Let V be all the elements w of $\mathcal{W}'$ that are not equivalent with w^*, i.e., such that there exists some query (β, γ) on which w and w^* differ. We will show next that G_n is the disjunction of events $G_n(w)$ over all $w \in V$.

If it were the case that for all $w \in V$, $G_n(w)$ does not hold for the current state, but G_n holds, then for all $w \in V$, $w \notin \mathcal{W}^{\kappa}_{min}$, so $\mathcal{W}^{\kappa}_{min}$ only contains elements equivalent with w^*; this implies $\mathrm{PO}(A, \mathcal{W}^{\kappa}_{min}) = \{\alpha^*\}$ (since we are assuming that only one alternative α^* is optimal in scenario w^*). But then the stopping criterion would hold, contradicting the definition of G_n. Hence G_n holds if and only if there exists $w \in V$ such that $G_n(w)$ holds. Therefore, $\Pr[G_n]$ is at most $\sum_{w \in V} \Pr[G_n(w)]$, where $\Pr[G_n]$ means the probability that event G_n happens at some point in the running of the algorithm, and similarly, $\Pr[G_n(w)]$.

We now consider the events of the form $G_n(w)$ in more detail, where $w \in V$. At Stage n, consider the total number K_w of queries in the sequence (i.e., among all queries asked up to this point) in which w and w^* differ. By the definition of Stage n (and since $w \in V$) we have $K_w \geq n$. Then $G_n(w)$ occurs if and only if G_n occurs and $\mathbf{X}_w^{w^*} = \mathit{mistakes}(w) - \mathit{mistakes}(w^*) \leq \kappa$.

For n greater than $\frac{\kappa}{1-2\rho}$, we can apply Proposition 3(ii) to give

$$\Pr[G_n(w)] \leq \exp\big(-2n(\frac{1}{2} - \rho - \frac{\kappa}{2n})^2\big).$$

Since this holds for all $w \in V$ we have

$$\Pr[G_n] \leq \sum_{w \in V} \Pr[G_n(w)] \leq |\mathcal{W}'| \exp\big(-2n(\frac{1}{2} - \rho - \frac{\kappa}{2n})^2\big).$$

Since $\lim_{n\to\infty} \exp\big(-2n(\frac{1}{2} - \rho - \frac{\kappa}{2n})^2\big) = 0$, we have:

$$\lim_{n\to\infty} \Pr[G_n] = 0$$

Let $f(q)$ be the highest n such that Stage n is guaranteed to happen after at most q queries if the algorithm has not stopped by then. Let H_q be the event that the algorithm has stopped after at most q queries. If not H_q, then Stage $f(q)$ was reached without fulfilling the stopping criterion, so if not H_q, then $G_{f(q)}$. Therefore we have:

$$1 - \Pr[H_q] \leq \Pr[G_{f(q)}]$$

Because of our assumption that each query is asked an infinite number of times if the algorithm never stops, we have $\lim_{q\to\infty} f(q) = \infty$. Therefore:

$$\lim_{q\to\infty}(1 - \Pr[H_q]) \leq \lim_{f(q)\to\infty} \Pr[G_{f(q)}] = 0$$

and therefore $\lim_{q\to\infty} H_q = 1$. So the probability that the algorithm terminates after at most q queries tends to 1 as q tends to infinity.

□

In the literature on termination of probabilistic programs, results of the same kind as the one shown by Theorem 2 have been called "almost sure termination", or "termination with probability one" (see for example [40]).

From Theorems 1 and 2 it immediately follows that the probability of the algorithm returning the correct optimal alternative tends to one as κ tends to infinity:

Corollary 2. *Along with the assumptions in Section 6.1, assume that in every infinite sequence of queries that can be generated by the algorithm, every possible query is asked an infinite number of times. Let $P_\kappa(\alpha^*)$ be the probability that the algorithm returns the correct optimal alternative α^*. Then $\lim_{\kappa\to\infty} P_\kappa(\alpha^*) = 1$.*

The condition, in Theorem 2 and Corollary 2, that every infinite sequence of asked queries contains each individual query an unlimited number of times, is included because of an example such as the following.

Example 2. *Consider a situation in which w agrees with w^* on all queries except that $w \cdot \beta > w \cdot \alpha^*$ while $w^* \cdot \beta < w^* \cdot \alpha^*$, where both w and w^* are in $\mathcal{W}'$. If the query between α^* and β is only asked a finite number of times, then after the last such query, mistakes(w) − mistakes(w^*) won't change. If this query is only asked a small number of times (such as less than κ), then most likely the set $\mathcal{W}^\kappa_{min}$ will continue forever to contain both w and w^* and so $\mathrm{PO}(A, \mathcal{W}^\kappa_{min})$ will contain both α^* and β, and the stopping criterion will never hold.*

Our algorithms in Sections 3 and 4 do not satisfy the condition that all queries are asked an unlimited number of times (if the algorithm does not terminate); however, they, and any other algorithm (i.e., method of generating the queries), can be easily modified to satisfy the condition, e.g., by very

occasionally cycling through all the possible queries. If the first such cycle is set to occur after the number of queries asked is more than the highest number of queries observed in our experiments, then our results from Section 5 remain exactly the same.

6.4. What happens when the decision-maker does not make any error

We now consider the case when the decision-maker always picks the correct alternative when queried, meaning that the error probability ρ is equal to 0.

Theorem 3. *If:*

- *The set $\mathcal{W}'$ contains an element that is equivalent to w^* (assumption on finite set $\mathcal{W}'$ from Section 6.1).*
- *Either each infinite sequence of generated queries contains each individual query an infinite number of times (assumption from Section 6.3), or each query is composed of two alternatives from $\mathrm{PO}(A, \mathcal{W}^{\kappa}_{min})$.*
- *When given a choice between two alternatives, the decision-maker always picks the one that has the higher utility for w^*.*

then:

- *The algorithm always terminates.*
- *The algorithm always returns the alternative α^* that maximises the utility for w^*.*

Proof: Let w_0 be an element of $\mathcal{W}'$ that is equivalent to w^*. Because the decision-maker does not make any error, *mistakes*(w_0) will remain at 0, and α^* will always be present in $\mathrm{PO}(A, \mathcal{W}^{\kappa}_{min})$. Therefore, if $\mathrm{PO}(A, \mathcal{W}^{\kappa}_{min})$ is reduced to a singleton, then the only alternative in $\mathrm{PO}(A, \mathcal{W}^{\kappa}_{min})$ will be α^*. Therefore, if the algorithm terminates, then it will return α^*.

It only remains to prove that the algorithm always terminates. Because the decision-maker does not make any error, then for any $w \in \mathcal{W}'$, the value of *mistakes*(w) (and of *mistakes*$(w) -$ *mistakes*(w_0), because *mistakes*$(w_0) = 0$) never decreases. Therefore, once a weights vector is removed from $\mathcal{W}^{\kappa}_{min}$, it cannot be part of $\mathcal{W}^{\kappa}_{min}$ again later.

Suppose that the algorithm never terminates. This means that there exists an alternative $\beta \neq \alpha^*$ that always stays in $\mathrm{PO}(A, \mathcal{W}^{\kappa}_{min})$. Let $w \in \mathcal{W}'$ be such that β is the alternative that maximises the utility for w. This means that *mistakes*(w) is always lesser or equal than κ. This means that the query (α^*, β) is asked at most κ times, and therefore that the assumption from Section 6.3 is not satisfied. So either the algorithm always terminates, or each query is composed of two alternatives from $\mathrm{PO}(A, \mathcal{W}^{\kappa}_{min})$. If the latter, then after at most $(p_{ini} - 1)(\kappa + 1)$ queries, with p_{ini} being the number of alternatives in $\mathrm{PO}(A, \mathcal{W}^{\kappa}_{min})$ at the start of the algorithm, all alternatives except $\alpha^\star$ will have been removed from $\mathrm{PO}(A, \mathcal{W}^{\kappa}_{min})$, fulfilling the stopping condition. Therefore the algorithm always terminates. □

Both query selection strategies that we studied in Section 4 only pick alternatives that are currently present in $\mathrm{PO}(A, \mathcal{W}^{\kappa}_{min})$, so our algorithm from Section 3 is covered by Theorem 3.

7. Conclusions and Discussion

We have described a novel, non-Bayesian, approach for interactive elicitation for a setting in which the user's answers are not completely reliable. Our approach is based on maintaining a set of plausible preference models and to reason about the alternatives that are optimal according to these preference models. We provide fast and effective methods for generating comparison queries. In our model the stopping criterion is defined so that the system finishes the interaction, and recommends an alternative α, when α is the optimal alternative in all the most plausible preference models. The notion of plausibility of a preference model is based on how close the answers from that model would be to the received answers. For computational reasons we focus attention on a finite approximation $\mathcal{W}'$ of the set of all preference models.

We have formally proven asymptotic correctness results on our algorithm, and on algorithms of similar form, including that the expected accuracy can be set to be arbitrarily close to 1 by increasing the κ parameter from the termination condition. Even with a low κ, our results show that the approach maintains good accuracy with reasonable lengths of interactions, and that it is very fast and suitable for real-time applications. We have also compared our approach with the Bayesian approach from [14], and our approach is very much faster, and with similar or perhaps slightly higher accuracy, and with similar number of queries required.

A variation of our approach that may further increase the accuracy, but at some computational cost, would be to update the finite set $\mathcal{W}'$ of preference models as the interaction progresses, so that the models become more densely populated in areas where they are most plausible.

Our stopping criterion, that a single alternative α in the set of alternatives A is optimal in all the most plausible user models $\mathcal{W}^{\kappa}_{min}$, is equivalent to the max regret of α (over $w \in \mathcal{W}^{\kappa}_{min}$) being zero. We can weaken this condition by instead enforcing that this minimum max regret is less than a small threshold ϵ, which will allow our method to be applied to some more complex situations and with different user models, and perhaps reducing the number of interactions with the decision-maker.

We have shown that our approach can deal with significant numbers of alternatives, and the number of alternatives does not appear to very strongly affect the computation time or performance of the algorithm. A natural further step would be to develop the approach for combinatorial problems, where there are an exponential number of alternatives. In this case, for each preference model w in the finite set $\mathcal{W}'$, we determine an optimal alternative α_w of the combinatorial problem with respect to the linear objective function given by w, and again the queries can be based on the possibly optimal elements with respect to the most plausible user models $\mathcal{W}^{\kappa}_{min}$.

Acknowledgements

This publication has emanated from research conducted with the financial support of Science Foundation Ireland under Grant number 12/RC/2289-P2, which is co-funded under the European Regional Development Fund; and with the support of the EU H2020 ICT48 project "TAILOR", under contract #952215. For the purpose of Open Access, the authors have applied a CC BY public copyright licence to any Author Accepted Manuscript version arising from this submission.

References

[1] J. S. Dyer, Interactive goal programming, Manage. Sci. 19 (1) (1972) 62–70. doi:10.1287/mnsc.19.1.62.
URL https://doi.org/10.1287/mnsc.19.1.62

[2] A. Salo, R. P. Hämäläinen, Preference ratios in multiattribute evaluation (prime)-elicitation and decision procedures under incomplete information, IEEE Trans. Syst. Man Cybern. Part A 31 (2001) 533–545.

[3] C. C. White III, A. P. Sage, S. Dozono, A model of multiattribute decisionmaking and trade-off weight determination under uncertainty, IEEE Trans. Syst. Man Cybern. 14 (2) (1984) 223–229. doi:10.1109/TSMC.1984.6313205.

[4] J. Blythe, Visual exploration and incremental utility elicitation, in: Eighteenth National Conference on Artificial Intelligence, American Association for Artificial Intelligence, USA, 2002, p. 526–532.

[5] C. Boutilier, A POMDP formulation of preference elicitation problems, in: Proceedings of AAAI02, 2002, pp. 239–246.

[6] U. Chajewska, L. Getoor, J. Norman, Y. Shahar, Utility elicitation as a classification problem, in: UAI '98: Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence, Morgan Kaufmann, 1998, pp. 79–88.

[7] U. Chajewska, D. Koller, R. Parr, Making rational decisions using adaptive utility elicitation, in: Proceedings of the Seventeenth National Conference on Artificial Intelligence and Twelfth Conference on on Innovative Applications of Artificial Intelligence, 2000, pp. 363–369.

[8] R. L. Keeney, H. Raiffa, Decisions with Multiple Objectives: Preferences and Value Trade-Offs, Wiley series in probability and mathematical statistics. Applied probability and statistics, Cambridge University Press, 1993.

[9] C. Boutilier, Computational Decision Support Regret-Based Models for Optimization and Preference Elicitation, in: Comparative Decision Making, Oxford University Press, 2013. doi:10.1093/acprof:oso/9780199856800.003.0041.

[10] N. Bourdache, P. Perny, Anytime algorithms for adaptive robust optimization with OWA and WOWA, in: J. Rothe (Ed.), Algorithmic Decision Theory - 5th International Conference, ADT 2017, Vol. 10576 of Lecture Notes in Computer Science, Springer, 2017, pp. 93–107.

[11] R. Price, P. R. Messinger, Optimal recommendation sets: Covering uncertainty over user preferences, in: AAAI, 2005, pp. 541–548.

[12] N. Bourdache, P. Perny, O. Spanjaard, Incremental elicitation of rank-dependent aggregation functions based on bayesian linear regression, in: Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19, 2019, pp. 2023–2029. doi:10.24963/ijcai.2019/280.

[13] I. Vendrov, T. Lu, Q. Huang, C. Boutilier, Gradient-based optimization for bayesian preference elicitation, Proceedings of the AAAI Conference on Artificial Intelligence 34 (06) (2020) 10292–10301. doi:10.1609/aaai.v34i06.6592.

[14] P. Viappiani, C. Boutilier, On the equivalence of optimal recommendation sets and myopically optimal query sets, Artificial Intelligence 286 (2020) 103328. doi:https://doi.org/10.1016/j.artint.2020.103328.
URL https://www.sciencedirect.com/science/article/pii/S0004370220300849

[15] N. Bourdache, P. Perny, O. Spanjaard, Bayesian preference elicitation for multiobjective combinatorial optimization, in: DA2PL 2020 - From Multiple Criteria Decision Aid to Preference Learning, Trento, Italy, 2020.
URL https://hal.archives-ouvertes.fr/hal-02979845

[16] N. Bourdache, P. Perny, O. Spanjaard, Bayesian preference elicitation for multiobjective combinatorial optimization, CoRR abs/2007.14778 (2020). arXiv:2007.14778.
URL https://arxiv.org/abs/2007.14778

[17] D. Sauré, J. P. Vielma, Ellipsoidal methods for adaptive choice-based conjoint analysis, Oper. Res. 67 (2) (2019) 315–338. doi:10.1287/opre.2018.1790.

[18] L. Adam, S. Destercke, Possibilistic preference elicitation by minimax regret, in: C. P. de Campos, M. H. Maathuis, E. Quaeghebeur (Eds.), Proceedings of the Thirty-Seventh Conference on Uncertainty in Artificial Intelligence, UAI 2021, Virtual Event, 27-30 July 2021, Vol. 161 of Proceedings of Machine Learning Research, AUAI Press, 2021, pp. 718–727.

[19] S. Teso, A. Passerini, P. Viappiani, Constructive preference elicitation by setwise max-margin learning, in: Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence, IJCAI 2016, 2016, pp. 2067–2073.

[20] R. Busa-Fekete, E. Hüllermeier, A survey of preference-based online learning with bandit algorithms, in: P. Auer, A. Clark, T. Zeugmann, S. Zilles (Eds.), Algorithmic Learning Theory - 25th International Conference, ALT 2014, Bled, Slovenia, October 8-10, 2014. Proceedings, Vol. 8776 of Lecture Notes in Computer Science, Springer, 2014, pp. 18–39. doi:10.1007/978-3-319-11662-4_3.
URL https://doi.org/10.1007/978-3-319-11662-4_3

[21] Y. Sui, M. Zoghi, K. Hofmann, Y. Yue, Advancements in dueling bandits, in: J. Lang (Ed.), Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI 2018, July 13-19, 2018, Stockholm, Sweden, ijcai.org, 2018, pp. 5502–5510. doi:10.24963/IJCAI.2018/776.
URL https://doi.org/10.24963/ijcai.2018/776

[22] J. Branke, S. Corrente, S. Greco, R. Słowiński, P. Zielniewicz, Using choquet integral as preference model in interactive evolutionary multiobjective optimization, European Journal of Operational Research 250 (3) (2016) 884–901. doi:https://doi.org/10.1016/j.ejor.2015.10.027.
URL https://www.sciencedirect.com/science/article/pii/S0377221715009534

[23] M. C. M. Troffaes, U. Sahlin, Imprecise swing weighting for multi-attribute utility elicitation based on partial preferences, in: A. Antonucci, G. Corani, I. Couso, S. Destercke (Eds.), Proceedings of the Tenth International Symposium on Imprecise Probability: Theories and Applications, Lugano, Switzerland, 10-14 July 2017, Vol. 62 of Proceedings of Machine Learning Research, PMLR, 2017, pp. 333–345.
URL http://proceedings.mlr.press/v62/troffaes17b.html

[24] C. Jansen, H. Blocher, T. Augustin, G. Schollmeyer, Information efficient learning of complexly structured preferences: Elicitation procedures and their application to decision making under uncertainty, Int. J. Approx. Reason. 144 (2022) 69–91. doi:10.1016/J.IJAR.2022.01.016.
URL https://doi.org/10.1016/j.ijar.2022.01.016

[25] S. Pourkhajouei, F. Toffano, P. Viappiani, N. Wilson, An efficient non-bayesian approach for interactive preference elicitation under noisy preference models, in: Z. Bouraoui, S. Vesic (Eds.), Symbolic and Quantitative Approaches to Reasoning with Uncertainty – 17th European Conference, ECSQARU 2023, Arras, France, September 19-22, 2023, Proceedings, Vol. 14294 of Lecture Notes in Computer Science, Springer, 2023, pp. 308–321.

[26] G. B. Hazen, Partial information, dominance, and potential optimality in multiattribute utility theory, Operations research 34 (2) (1986) 296–310.

[27] M. Gelain, M. S. Pini, F. Rossi, K. B. Venable, T. Walsh, Elicitation strategies for soft constraint problems with missing preferences: Properties, algorithms and experimental studies, Artif. Intell. 174 (3-4) (2010) 270–294.

[28] N. Benabbou, P. Perny, Incremental weight elicitation for multiobjective state space search, in: Proceedings of Twenty-Ninth AAAI Conference on Artificial Intelligence, 2015, pp. 1093–1099.

[29] N. Wilson, A. Razak, R. Marinescu, Computing possibly optimal solutions for multi-objective constraint optimisation with tradeoffs, in: Proc. IJCAI-2015, 2015.

[30] I. Levi, On indeterminate probabilities, Journal of Philosophy 71 (13) (1974) 391–418. doi:10.2307/2025161.

[31] L. V. Utkin, T. Augustin, Powerful algorithms for decision making under partial prior information and general ambiguity attitudes, in: F. G. Cozman, R. Nau, T. Seidenfeld (Eds.), ISIPTA '05, Proceedings of the Fourth International Symposium on Imprecise Probabilities and Their Applications, Carnegie Mellon University, Pittsburgh, PA, USA, July 20-23 2005, SIPTA, 2005, pp. 349–358.

[32] C. Jansen, G. Schollmeyer, T. Augustin, Quantifying degrees of e-admissibility in decision making with imprecise probabilities, in: T. Augustin, F. G. Cozman, G. Wheeler (Eds.), Reflections on the Foundations of Probability and Statistics: Essays in Honor of Teddy Seidenfeld, Springer International Publishing, Cham, 2022, pp. 319–346.

doi:10.1007/978-3-031-15436-2_13.
URL https://doi.org/10.1007/978-3-031-15436-2_13

[33] F. Toffano, N. Wilson, Minimality and comparison of sets of multi-attribute vectors, Autonomous Agents and Multi-Agent Systems 36 (2) (2022) 1–66.

[34] S. Zionts, J. Wallenius, An interactive programming method for solving the multiple criteria problem, Management science 22 (6) (1976) 652–663.

[35] R. E. Steuer, E.-U. Choo, An interactive weighted tchebycheff procedure for multiple objective programming, Mathematical programming 26 (3) (1983) 326–344.

[36] C. Boutilier, R. Patrascu, P. Poupart, D. Schuurmans, Constraint-based optimization and utility elicitation using the minimax decision criterion, Artif. Intell. 170 (8-9) (2006) 686–713. doi:10.1016/J.ARTINT.2006.02.003.
URL https://doi.org/10.1016/j.artint.2006.02.003

[37] P. Viappiani, C. Boutilier, Regret-based optimal recommendation sets in conversational recommender systems, in: L. D. Bergman, A. Tuzhilin, R. D. Burke, A. Felfernig, L. Schmidt-Thieme (Eds.), Proceedings of the 2009 ACM Conference on Recommender Systems, RecSys 2009, New York, NY, USA, October 23-25, 2009, ACM, 2009, pp. 101–108.

[38] F. Toffano, P. Viappiani, N. Wilson, Efficient exact computation of set-wise minimax regret for interactive preference elicitation, in: AAMAS '21: 20th International Conference on Autonomous Agents and Multiagent Systems, 2021, pp. 1326–1334. doi:10.5555/3463952.3464105.

[39] W. Hoeffding, Probability inequalities for sums of bounded random variables, Journal of the American Statistical Association 58 (301) (1963) 13–30. doi:10.1080/01621459.1963.10500830.

[40] J. Esparza, A. Gaiser, S. Kiefer, Proving termination of probabilistic programs using patterns, in: P. Madhusudan, S. A. Seshia (Eds.), Computer Aided Verification - 24th International Conference, CAV 2012, Berkeley, CA, USA, July 7-13, 2012 Proceedings, Vol. 7358

of Lecture Notes in Computer Science, Springer, 2012, pp. 123–138. doi:10.1007/978-3-642-31424-7_14.
URL https://doi.org/10.1007/978-3-642-31424-7_14