



POLITECNICO
MILANO 1863

SCUOLA DI INGEGNERIA INDUSTRIALE
E DELL'INFORMAZIONE

Multi-Dimensional Reward Learning from Preference Feedback

TESI DI LAUREA MAGISTRALE IN
COMPUTER SCIENCE AND ENGINEERING - INGEGNERIA IN-
FORMATICA

Author: **Leonardo Bianconi**

Student ID: 232004

Advisor: Prof. Alberto Maria Metelli

Co-advisors: Simone Drago, Marco Mussi

Academic Year: 2024-25

Abstract

Reinforcement Learning (RL) is a powerful framework for solving sequential decision-making problems by maximizing the accumulated sum of a suitable scalar reward signal. The choice of the reward is fundamental in obtaining good performances on the task, however, defining such reward is often challenging, requiring lots of task-specific knowledge and often needing to account for different underlying objectives. Several approaches try to overcome this criticality. Preference-based Reinforcement Learning (PbRL) algorithms learn to solve the task by asking a human expert for preferences over examples of solutions for the task (called trajectories), instead of requiring a reward; usually, this is done by learning a surrogate reward (called *reward model*) from comparison data, and then utilize standard RL techniques. An additional challenge arises when the agent has to handle multiple, and often contrasting, objective. In the literature, Multi-Objective Reinforcement Learning (MORL) enables the specification of multi-dimensional rewards, allowing to effectively learn several policies for different combinations of the objectives. When translating the multi-objective problem to PbRL, this corresponds to an expert not being able to provide an explicit preference since the trajectories may be incomparable, meaning that neither is better than the other w.r.t. all the objectives. This possibility enables learning a reward model that encodes the various dimensions of the task, ultimately obtaining a MORL problem instance. In this work, we present the novel setting in which an algorithm learns a multi-dimensional reward model from preference data. We formalize this setting, define the desiderata that a preference model needs to satisfy, and present an important negative result concerning the optimization procedure of such models. We then define three choices of models that extend the Bradley-Terry (BT) model (a probabilistic model adopted ubiquitously in standard reward modeling) and that comply, in part or completely - accounting for optimization guarantees - with the aforementioned desiderata. Finally, we validate these preference models through various experiments, focusing on their ability of learning from small amount of data and from non-optimal experts.

Keywords: Reward Modeling, Preference-based Reinforcement Learning, Multi-Objective Reinforcement Learning, Bradley-Terry Model.

Abstract in lingua italiana

Il Reinforcement Learning (RL) è un potente framework per risolvere problemi di decisione sequenziale massimizzando la somma accumulata di un opportuno segnale di ricompensa scalare. La scelta della ricompensa è fondamentale per ottenere buone prestazioni sul problema; tuttavia, definirla è spesso difficile, richiedendo molta conoscenza del dominio e la necessità di considerare diversi obiettivi sottostanti. Diversi approcci cercano di superare questa criticità. Gli algoritmi di Preference-based Reinforcement Learning (PbRL) imparano a risolvere il compito chiedendo a un esperto umano di esprimere preferenze tra esempi di soluzioni del problema (chiamate traiettorie), invece di richiedere una ricompensa esplicita; solitamente, ciò avviene apprendendo una ricompensa surrogata (chiamata reward model) da queste preferenze, per poi utilizzare tecniche standard di RL. Una sfida aggiuntiva emerge quando l'agente deve gestire obiettivi multipli, spesso in contrasto tra loro. In letteratura, il Multi-Objective Reinforcement Learning (MORL) consente la specificazione di ricompense multidimensionali, permettendo di apprendere efficacemente diverse politiche per diverse combinazioni degli obiettivi. Quando si traduce il problema multi-obiettivo in PbRL, ciò corrisponde a un esperto che non è in grado di fornire una preferenza poiché le traiettorie possono essere incomparabili, ossia, nessuna è migliore dell'altra rispetto a tutti gli obiettivi. Questa possibilità consente di apprendere un reward model che codifica le varie dimensioni del problema, ottenendo un'istanza di problema di MORL. In questo lavoro presentiamo un nuovo scenario in cui un algoritmo apprende un reward model multidimensionale a partire da preferenze. Formalizziamo questo setting, definiamo i requisiti (desiderata) che il modello di preferenza deve soddisfare e presentiamo un importante risultato negativo riguardante l'ottimizzazione di tali modelli. Definiamo poi tre scelte di modelli che estendono il modello di Bradley-Terry (BT) — un modello probabilistico ampiamente adottato in reward modeling classico — e che soddisfano, in parte o completamente (considerando le garanzie di ottimizzazione), i suddetti requisiti. Infine, validiamo questi modelli tramite diversi esperimenti, focalizzandoci sulla loro capacità di apprendere da dataset limitati e da esperti non ottimali.

Parole chiave: Reward Modeling, Preference-based Reinforcement Learning, Multi-

Objective Reinforcement Learning, Modello di Bradley-Terry.

Contents

Abstract	i
Abstract in lingua italiana	iii
Contents	v
1 Introduction	1
1.1 Motivation	1
1.2 Goal	3
1.3 Original Contributions	3
1.4 Thesis Structure	4
2 Preliminaries	5
2.1 Convex Functions	5
2.2 Statistical Divergences	6
2.3 Order Relations	6
2.3.1 Pre-orders, Orders, Equivalence Relations	6
2.3.2 Order Dimension, Width	7
2.4 Reinforcement Learning	9
2.5 Preference-based Reinforcement Learning	12
2.6 Multi-Objective Reinforcement Learning	17
3 Related Works	21
3.1 PbRL Methodologies for Indifference and Incomparability	21
3.2 Computational Issues in Multi-Objective PbRL	23
4 Problem Setting and Reward Model Desiderata	27
4.1 Why the Need for Multi-Dimensional Reward Models	27
4.2 Problem Setting and Model Desiderata	28
4.2.1 Problem Setting	28

4.2.2	Desiderata for a Reward Model	29
4.3	Non-Convexity of Negative Log-Likelihood	32
5	Models' Derivations	35
5.1	Extending the Bradley-Terry Model	35
5.1.1	Derivation of Model A	36
5.1.2	Derivation of Model B	41
5.1.3	Derivation of Model C	44
6	Experiments	49
6.1	Implementation Details	49
6.2	Preliminary Validation on Synthetic Datasets	50
6.3	Multi-Objective Gridworld	54
7	Conclusions	67
7.1	Future Works	67
	Bibliography	69
	List of Figures	77
	List of Tables	79
	Acknowledgements	81

1 | Introduction

Reinforcement Learning [RL, 55] is a well-known paradigm of Machine Learning [ML, 7] in which an agent learns to control a dynamic environment by acting on it. "Controlling", in this sense, means learning a decision-making strategy (namely, a *policy*) that maximizes a scalar reward signal that is given to the agent by the environment, and that encodes the notion of the goal for the task. Due to its strong theoretical foundations, RL has become a very attractive tool to solve sequential decision-making problems from data, and has already seen many successful applications, starting from playing board games such as backgammon [56] and Go [50] at world champions' level, and spanning to robotics [32] and medical applications [59, 72].

1.1. Motivation

In real-life applications, a critical and not straightforward task is engineering the reward function: infinite choices exist, and, in the majority of cases, many objectives, often conflicting with each other, need to be encoded in the reward. Moreover, the agent's learned policy is very sensible to small perturbations of the reward [43], possibly giving rise to *reward hacking* issues, i.e., the agent may try to maximize the given reward by "hacking" it and not pursuing the real goal of the task [3]. Due to these criticalities, several alternatives have been proposed in the literature.

A framework that tries to cope with such problem is Preference-based Reinforcement Learning [PbRL, 68]. PbRL approaches this problem by changing the paradigm of interaction between the agent and the environment: the reward is substituted by a human expert capable of labeling pairs of interactions (called trajectories) between the agent and the environment, providing a preference between the two to the agent. In recent years, PbRL has seen great advances in both theoretical works [49] and practical applications [27].

Another framework directly dealing with the reward engineering problem is Multi-Objective Reinforcement Learning [MORL, 25], a paradigm of RL in which the reward signal is

multidimensional, therefore taking into account diverse objectives for the task. MORL algorithms make it possible for the agent to learn a family of behaviors for the agent, each being optimal for a certain relative importance between the objectives. This approach is surely more generic than classic RL, however, it still requires a correctly engineered reward function, that has to be chosen from infinite possibilities.

Learning from preferences is again problematic in the case of multiple underlying task objectives, since the expert may be asked to label a pair of trajectories in which one trajectory is better than the other with respect to one objective, and vice versa for some other objective. This is a scenario that has been widely overlooked in previous literature.

The main approach utilized by PbRL algorithms is to separate the problem in two steps: (a) learning a *reward model* from preference feedback, i.e., a numerical, surrogate reward signal to then (b) learn a policy that is optimal with respect to this learned reward model. The most employed model and de-facto standard way to learn a reward model is the *Bradley-Terry* [BT, 10] *model* [10], a probabilistic approach that models the expert behavior in selecting preferences for trajectory pairs.

Recently, [19] analyzed the computational complexity of representing order relations with multi-dimensional utility signals in PbRL, conjecturing that the problem of assessing whether a policy is Pareto-optimal with respect to all other policies is NP-hard. This result motivates shifting to employing numerical reward signals that, at the price of introducing a bias, provide more friendly computational properties. The idea of learning a multi-dimensional reward model from preference feedback with the possibility of two trajectories to be *incomparable* by the expert remains, to this day, only an interesting unexplored research direction [68].

Being able to learn a family of Pareto-optimal policies in multi-objective settings when only comparison feedback from an expert is available would be of paramount importance as it would simplify the reward engineering problem in multi-objective problems, that are very frequent in practical applications. A clear example is autonomous driving [71, 26], in which a human would like to optimize many contrasting objectives, such as quickly arriving at the destination, preserving safety for herself and the other road users, not breaking traffic laws, maximizing comfort for the passengers, and many more. However, it is far from trivial for a human to numerically encode all those objectives in a vector, while it is much simpler to assess whether two trajectories (i.e., two travels in an autonomous vehicle) are one preferred to the other or the two maximize contrasting objectives.

Large Language Models [LLMs, 8], whose popularity and usefulness in everyday life is constantly increasing, utilize PbRL algorithms in their training process, during the final

alignment phase [42]. This phase has the objective of avoiding unwanted behaviors from the LLM, while aligning with desirable objectives such as correctness, completeness, coherence, brevity or harmlessness of the LLM responses. All these objectives are in some way contrasting with some other (for example, completeness and brevity), however, experts are only able to give a strict preference between trajectories (i.e., LLM responses to a query) and the incomparable ones are simply discarded.

1.2. Goal

The goal of this research is to address the problem of learning multi-dimensional reward models in the case of an expert labeler able to, given a pair of trajectories, either express a preference of one with respect to the other or declare the pair as *incomparable*. Precisely, the objectives are formalizing the problem, formalizing what a probabilistic expert model should achieve in such setting and then designing suitable expert models that are the most general (for example, that can effortlessly adapt to any dimensionality of the problem at hand), that reduce to already de-facto standard models in the case of scalar rewards and that have guarantees on the optimization procedure.

1.3. Original Contributions

The contributions of this work can be summarized as follows:

- We introduce a novel framework in which an agent interacts with an MDP without explicit reward function, but in presence of an expert capable of, given a trajectory pair, expressing feedbacks belonging to four distinct classes: two classes for strict preference, one for indifference and one for incomparability between the pair. The expert is modeled probabilistically.
- We define the *rational multi-objective expert model*, i.e., a set of properties that any expert acting on the difference of utility score $U(\tau_1) - U(\tau_2)$ of the two trajectories must satisfy. These conditions are expressed, without any assumption on the structure of the probability function, on the 4-categorical probability distribution of the label.
- We present an important negative result: any model complying with the rational multi-objective expert model definition has non-convex negative log-likelihood in the general case. We describe the implications of this result and the way this impacts the subsequent design choices for expert models.

- We present three choices of expert model (Model A, Model B and Model C), illustrating the choices made, in each case, regarding the trade-off between expressivity of the model (in terms of degree of compliance with the aforementioned definition) and optimization guarantees. Moreover, we validate these expert models through various experiments, analyzing the optimization results on small to considerable-size datasets and conducting robustness studies in which the expert behaves sub-optimally.

1.4. Thesis Structure

We start by revising, in Chapter 2, useful concepts from various fields, needed to understand the setting of the problem and our contributions. Then, in Chapter 3, we present literature works that are related to our contributions, highlighting the differences with our work. In Chapter 4, we formalize the setting of our problem and present the desiderata of a reward model in this setting, discussing a negative result concerning any model complying with such desiderata. Then, in Chapter 5, we propose three choices of models complying, totally or partly, with the aforementioned desiderata, highlighting the design choices made. In Chapter 6, we validate and analyze these models through experiments on various datasets. Finally, in Chapter 7, we draw the conclusions of our work, and present some possible future working directions.

2 | Preliminaries

In this chapter, we present the basic notions required to understand the contents of the following chapters, and in particular to understand the setting of the problem and our contributions. Firstly, in Section 2.1, we introduce the concept of convex function, then, in Section 2.2, we present two important statistical distances. In Section 2.3, we revise useful notions from order relation theory, then, in Section 2.4, we introduce Reinforcement Learning; in Section 2.5, we present some concepts from its specialization field Preference-based Reinforcement Learning. Finally, in Section 2.6, we introduce key notions of Multi-Objective Reinforcement Learning.

2.1. Convex Functions

A set \mathcal{S} is *convex* if the line segment between any two points in \mathcal{S} lies in \mathcal{S} , i.e., for any $x, y \in \mathcal{S}$ and any $\lambda \in [0, 1]$ we have:

$$\lambda x + (1 - \lambda)y \in \mathcal{S}. \quad (2.1)$$

A function $f : \mathbb{R}^n \rightarrow \mathbb{R}$ is *convex* in its arguments if $\text{dom}(f)$ is a convex set and if, for all $\mathbf{x}, \mathbf{y} \in \text{dom}(f)$, and $\lambda \in [0, 1]$, the following property holds:

$$f(\lambda \mathbf{x} + (1 - \lambda)\mathbf{y}) \leq \lambda f(\mathbf{x}) + (1 - \lambda)f(\mathbf{y}). \quad (2.2)$$

Geometrically, it means that the line segment connecting $(x, f(x))$ to $(y, f(y))$ lies above the graph of the function f . A function f is said *concave* if $-f$ is convex. As an example consider any linear affine function $f : \mathbb{R}^d$, defined as $f(\mathbf{x}) = \mathbf{a}^\top \mathbf{x} + b$: it can be trivially proved that such function is both convex and concave (intuitively, property in Equation 2.2 holds as equality, since the line segment connecting any two points overlaps exactly with the graph of the f).

2.2. Statistical Divergences

Given two discrete probability distributions $P(x)$ and $Q(x)$, $x \in \mathcal{X}$, the *Kullback-Leibler* [KL, 33] *divergence* of P with respect to Q , denoted as $D_{KL}(P \parallel Q)$ is defined as:

$$D_{KL}(P \parallel Q) := \sum_{x \in \mathcal{X}} P(x) \log \frac{P(x)}{Q(x)}. \quad (2.3)$$

The Kullback-Leibler divergence is an asymmetric measure of distance between the two probability distributions, and, informally speaking, quantifies how inefficient it is, on average, to approximate a (real) distribution P with an approximator distribution Q .

The *Jensen-Shannon divergence* between two probabilities distributions $P(x)$ and $Q(x)$ is defined as:

$$D_{JS}(P \parallel Q) := \frac{1}{2}D_{KL}(P \parallel M) + \frac{1}{2}D_{KL}(Q \parallel M) \quad (2.4)$$

where $M := \frac{P+Q}{2}$ is a *mixture* distribution of P and Q . This measure has the advantage to be symmetric with respect to P and Q . We employ this metric in Chapter 6.

2.3. Order Relations

2.3.1. Pre-orders, Orders, Equivalence Relations

Given two sets \mathcal{A} and \mathcal{B} , a binary relation between \mathcal{A} and \mathcal{B} is defined as a subset R of the Cartesian product between the two sets (i.e., product set). In mathematical terms, $R \subseteq \mathcal{A} \times \mathcal{B}$. From now on, we will consider, for later convenience, $\mathcal{A} = \mathcal{B} = \mathcal{X}$. We shall also make use of the infix notation: we will denote $(x, y) \in R$ as xRy . We can define some properties that a relation $R \subseteq \mathcal{X} \times \mathcal{X}$ may satisfy:

- (a) *reflexivity*: $\forall x \in \mathcal{X} : xRx$
- (b) *transitivity*: $\forall x, y, z \in \mathcal{X} : xRy \wedge yRz \implies xRz$
- (c) *symmetry*: $\forall x, y \in \mathcal{X} : xRy \implies yRx$
- (d) *antisymmetry*: $\forall x, y \in \mathcal{X} : xRy \wedge yRx \implies x = y$

A relation $\preceq_{\mathcal{X}} \subseteq \mathcal{X} \times \mathcal{X}$ is called a *pre-order* if it is reflexive and transitive. A relation $\preceq_{\mathcal{X}} \subseteq \mathcal{X} \times \mathcal{X}$ is called a *partial order* if it is a pre-order and it is also antisymmetric. A (pre-)order is called *total* if $\forall x, y \in \mathcal{X} : x \preceq_{\mathcal{X}} y \vee y \preceq_{\mathcal{X}} x$. We say that two elements $x, y \in \mathcal{X}$ are *equivalent* with respect to $\preceq_{\mathcal{X}}$ if $x \preceq_{\mathcal{X}} y \wedge y \preceq_{\mathcal{X}} x$, and we denote it by $x \simeq_{\mathcal{X}} y$. Similarly, we say that x and y are *incomparable* (and denote it by $x \parallel_{\mathcal{X}} y$) if

neither $x \preceq_{\mathcal{X}} y$ nor $y \preceq_{\mathcal{X}} x$ (otherwise, we say that x and y are *comparable*). We notice that a pre-order allows two elements to be equivalent while still being distinct, while this is not true in an order.

The aforementioned notion of equivalence can be formalized by defining the *equivalence relation* as a reflexive, symmetric and transitive relation between elements of \mathcal{X} . It can be seen that $\succsim_{\mathcal{X}}$ is indeed an equivalence relation. Given an element $x \in \mathcal{X}$, we denote its *equivalence class* as $[x] := \{y \in \mathcal{X} : y \succsim_{\mathcal{X}} x\}$. We define the *quotient set* $\mathcal{X} / \succsim_{\mathcal{X}}$ as the set of all the equivalence classes created by $\succsim_{\mathcal{X}}$ over \mathcal{X} . $\succsim_{\mathcal{X}}$ is an equivalence relation that induces a partial order over the quotient set $\mathcal{X} / \succsim_{\mathcal{X}}$, i.e., $[x] \preceq_{\mathcal{X} / \succsim_{\mathcal{X}}} [y]$ if $x \preceq_{\mathcal{X}} y$.

As a simple example, one can think of the "less than or equal to" relation between natural numbers (\mathbb{N}). This is an order relation, since it is (trivially) reflexive, antisymmetric and transitive; moreover, the order is total since each pair of natural numbers is comparable in terms of which is greater. Another, important example is the *component-wise* (or Pareto) partial order between real vectors $\mathbf{u}, \mathbf{v} \in \mathbb{R}^d$, defined as follows: $\mathbf{u} \preceq \mathbf{v} \iff \forall i \in \llbracket d \rrbracket : u_i \leq v_i$, and $\mathbf{u} \prec \mathbf{v} \iff \forall i \in \llbracket d \rrbracket : u_i \leq v_i \wedge \exists j \in \llbracket d \rrbracket : u_j < v_j$.

Other examples on small sets are depicted in Figure 2.1.

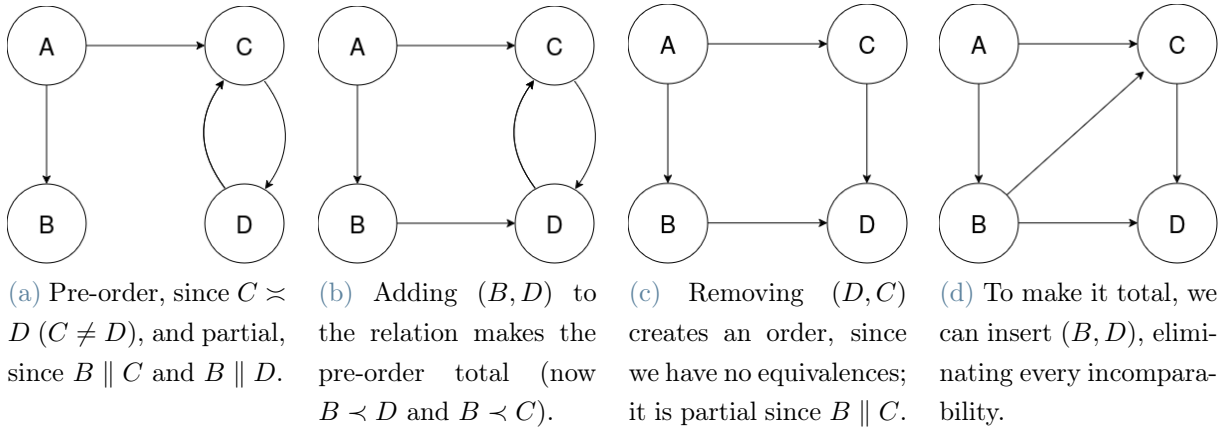


Figure 2.1: Graphs of order relations; nodes represent elements $x \in \mathcal{X}$, and arcs $x \rightarrow y$ are drawn if $(x, y) \in \preceq_{\mathcal{X}}$ (reflexive and transitive arcs are omitted for clarity).

2.3.2. Order Dimension, Width

In this Section, we define two important properties of any order relation: *order dimension* and *width*.

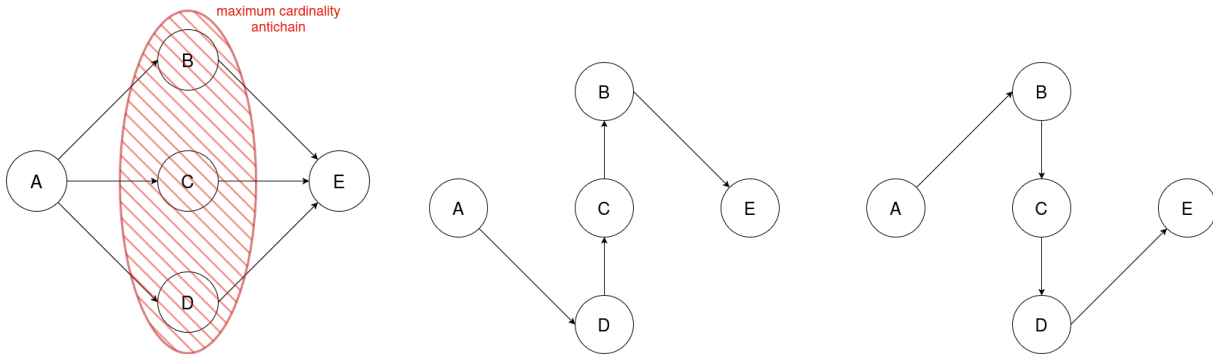
Given an order relation $\preceq_{\mathcal{X}} \subseteq \mathcal{X} \times \mathcal{X}$ and a total order relation $\leq_{\mathcal{X}} \subseteq \mathcal{X} \times \mathcal{X}$, then $\leq_{\mathcal{X}}$ is said to be a *linear extension* of $\preceq_{\mathcal{X}}$ if $\preceq_{\mathcal{X}} \subseteq \leq_{\mathcal{X}}$ (i.e., $x \preceq_{\mathcal{X}} y \implies x \leq_{\mathcal{X}} y$). A

realizer of an order relation $\preceq_{\mathcal{X}}$ is a set of total orders $\{\preceq_{\mathcal{X},i}\}_{i \in [d]}$ s.t. $\bigcap_{i \in [d]} \preceq_{\mathcal{X},i} = \preceq_{\mathcal{X}}$. This concept is fundamental in defining an intrinsic property of any order relation: the *order dimension* [20, 58]. The order dimension of an order $\preceq_{\mathcal{X}}$, denoted as $\dim(\preceq_{\mathcal{X}})$, is defined as the least cardinality of a realizer of that order (i.e., $d = \min\{d \in \mathbb{N} : \exists \{\preceq_{\mathcal{X},i}\}_{i \in [d]} \text{ s.t. } \bigcap_{i \in [d]} \preceq_{\mathcal{X},i} = \preceq_{\mathcal{X}}\}$). In the case of a total order relation $\preceq_{\mathcal{X}}$, trivially $\dim(\preceq_{\mathcal{X}}) = 1$. We can extend the concept of order dimension to pre-orders by defining it as the order dimension of the partial order induced over the quotient set, i.e., $\dim(\preceq_{\mathcal{X}}/\sim_{\mathcal{X}})$.

An *antichain* is a subset of \mathcal{X} such that all elements in the subset are incomparable with each other. The *width* of an order relation is the maximum cardinality of an antichain (i.e., $\text{width}(\preceq_{\mathcal{X}}) = \max\{|\mathcal{Y}| : \mathcal{Y} \subseteq \mathcal{X} \text{ s.t. } \forall x, y \in \mathcal{Y} : x \neq y \implies x \parallel_{\mathcal{X}} y\}$).

It is known that $\dim(\preceq_{\mathcal{X}}) \leq \text{width}(\preceq_{\mathcal{X}})$ [18]. From a computational perspective, it is known that verifying whether the order dimension is at most k is NP-hard for $k \geq 3$ [23, 70]; furthermore, unless NP=ZPP, there exists no polynomial-time algorithm to approximate the order dimension with a factor of $O(|\mathcal{X}|^{1-\epsilon})$, for every $\epsilon > 0$ [14]. Calculating the width of an order relation is, instead, a problem solvable in polynomial time, and a realizer of size $|\text{width}(\preceq_{\mathcal{X}})|$ can be found solving a minimum path cover problem in $O(|\mathcal{X}|^3)$.

An illustrative example of width and dimension of a partial order is depicted in Figure 2.2.



(a) This is the graph of a partial order $\preceq_{\mathcal{X}}$. We can see that $B \parallel C \parallel D$, therefore forming an antichain of length 3; giving $\text{width}(\preceq_{\mathcal{X}}) = 3$.

(b) It is trivial to verify that the intersection of these two total orders coincides with the initial order $\preceq_{\mathcal{X}}$. This is indeed a realizer, and one of minimum cardinality; therefore $\dim(\preceq_{\mathcal{X}}) = 2$.

Figure 2.2: Example of partial order, antichain and realizer.

2.4. Reinforcement Learning

Reinforcement Learning [RL, 55] is a very well known paradigm of Machine Learning [ML, 7] in which an agent is placed inside a dynamic environment, and its goal is to learn to control such environment. The agent does so by learning a behavioral policy, i.e., a mapping between a particular state of the environment and the action the agent needs to perform in that state. The ultimate objective of the agent is to maximize a scalar reward function, that can be observed by the agent during the learning process.

Each practical instance of a sequential decision-making problem needs to be tackled by modeling the environment and the interaction protocol between the learning agent and the environment. The modeling choice for the environment that is the most common in the literature and most frequently used in real-life applications is the Markov Decision Process [MDP, 46]. In this work, we focus on its finite-horizon version, in which each interaction between the agent and the MDP is finite and has the same length in terms of temporal timesteps. In the following, two important models that will be used extensively throughout the contributions section are defined.

Finite-horizon Markov Decision Process without Reward (MDP\R) A finite-horizon Markov Decision Process Without Reward [MDP\R, 1] is a tuple $\mathcal{M} := \langle \mathcal{S}, \mathcal{A}, H, P, \mu_0 \rangle$ where:

- \mathcal{S} is the (finite) set of states.
- \mathcal{A} is the (finite) set of actions.
- $H \in \mathbb{N}$ is the time horizon: after this amount of timesteps, the interaction ends.
- P is the state transition probability function $P : \mathcal{S} \times \mathcal{A} \times \llbracket H \rrbracket \rightarrow \Delta(\mathcal{S})$, $P(s'|s, a, t)$ models the probability of going from state s to s' playing action a at timestep t .
- $\mu_0 \in \Delta(\mathcal{S})$ is the initial state distribution, $\mu_0(s)$ specifies the probability that the environment is initialized at state s .

Finite-horizon Markov Decision Process (MDP) A finite-horizon Markov Decision Process [MDP, 46] is a tuple $\mathcal{M} := \langle \mathcal{S}, \mathcal{A}, H, P, R, \mu_0 \rangle$ where:

- $\langle \mathcal{S}, \mathcal{A}, H, P, \mu_0 \rangle$ is a finite-horizon MDP\R.
- R is the scalar reward function $R : \mathcal{S} \times \mathcal{A} \times \llbracket H \rrbracket \rightarrow \mathbb{R}$, $R(s, a, t)$ specifies the reward that the agent receives from the environment by playing action a in state s at timestep t .

A *trajectory* is an ordered sequence of state-action pairs $\tau := (s_h, a_h)_{h \in [H]}$. We denote the space of all possible trajectories as $\mathcal{T} \subseteq (\mathcal{S} \times \mathcal{A})^H$.

Markov property The MDP model presents one important property, namely the Markov property for stochastic processes. A stochastic process X_t is said to be *Markovian* if and only if:

$$\mathbb{P}(X_{t+1} = j | X_t = i, X_{t-1} = k_{t-1}, \dots, X_1 = k_1, X_0 = k_0) = \mathbb{P}(X_{t+1} = j | X_t = i). \quad (2.5)$$

In other words, the current state of the process captures all the information about the past, and the past history of the process may be "forgotten". One can easily verify that this property holds for MDPs, since both the transition model P and the reward model R do not depend on previously visited states and played actions.

In RL, the agent interacts with an environment, usually modelled as MDP, to maximize the cumulative reward. The interaction proceeds as follows: at each timestep t , the agent receives the current state s_t from the environment, "plays" an action $a_t \in \mathcal{A}$ and, subsequently, receives the reward r_t from the environment. The whole interaction protocol is sketched in Figure 2.3.

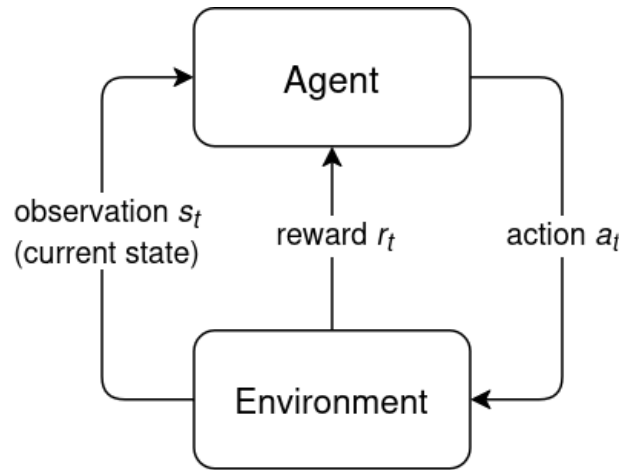


Figure 2.3: Sketch of the interaction protocol between an RL agent and the environment.

Policy Given a finite-horizon MDP $\langle \mathcal{S}, \mathcal{A}, H, P, R, \mu_0 \rangle$ a (Markovian) *policy* is a collection $\pi = (\pi_h)_{h \in [H]}$ of probability distributions $\pi_h(a|s) = \mathbb{P}(a|s, h)$ over the action space \mathcal{A} , given the state s , at timestep h .

Value functions Given a policy π we define the *state-value function* as the expected return obtained starting from s at timestep t , and following policy π :

$$V_t^\pi(s) = \mathbb{E} \left[\sum_{h=t+1}^H r_h | \pi, s_t = s \right]. \quad (2.6)$$

Similarly, given a policy π , we define the *action-value function* as the expected return obtained from s at timestep t , playing $a_t = a$ and then following policy π :

$$Q_t^\pi(s, a) = \mathbb{E} \left[\sum_{h=t+1}^H r_h | \pi, s_t = s, a_t = a \right]. \quad (2.7)$$

Note: the expectation in Equations 2.6 and 2.7 is computed with respect to the stochasticity of the policy and the environment.

Solving a finite-horizon MDP The optimal state-value function $V^*(s)$ is the maximum value function, at $t = 0$, over all policies:

$$V^*(s) = \max_{\pi} V_0^\pi(s). \quad (2.8)$$

The ultimate goal of an agent in an MDP is to learn an optimal policy π^* , i.e., one maximizing the value function in each state:

$$\pi^* \in \arg \max_{\pi} \{V_0^\pi(s)\}. \quad (2.9)$$

Bellman Optimality Equation [6] The Bellman Optimality Equation for V^* is an equation expressing the fact that the value of a state under an optimal policy must equal the expected return for the best action from that state:

$$V_t^*(s) = \max_a Q_t^*(s, a) = \max_a \left\{ R(s, a, t) + \sum_{s' \in \mathcal{S}} P(s'|s, a, t) V_t^*(s') \right\}. \quad (2.10)$$

From V^* to π^* It can be proven that, given an MDP, there always exists a deterministic, Markovian optimal policy, and it can be easily obtained starting from the optimal value function $V^*(s)$ by selecting, in each state, the action maximizing the expected value of

the next state:

$$\pi_t^*(a|s) = \arg \max_a \left\{ R(s, a, t) + \sum_{s' \in \mathcal{S}} P(s'|s, a, t) V_t^*(s') \right\}. \quad (2.11)$$

The main approaches in solving an MDP are (a) finding a fixed point of the Bellman Optimality Equation (Equation 2.10), for example using Dynamic Programming (DP) approaches such as Value Iteration (VI), Policy Iteration (PI), any Generalized Policy Iteration (GPI) algorithm or using RL algorithms such as Q-learning [64]; (b) using optimization formulations of the MDP problem, solve Linear Programs (LP) or convex programs to find an optimal policy. For the specific case of the finite-horizon, Backward Induction (BI) is usually adopted to obtain the optimal value function in a finite number of steps, then an optimal policy can be derived using Equation 2.11.

2.5. Preference-based Reinforcement Learning

In this section, we introduce a specialization of Reinforcement Learning, namely Preference-based Reinforcement Learning. We start by introducing one of the major issues concerning the typical RL paradigm: the *reward engineering* phase; then, we define a possible solution to such problem and describe modeling choices, objectives and methodologies of Preference-based Reinforcement Learning.

Formalizing a problem environment as an MDP requires engineering a suitable reward function R . It has been noted [43] that the choice of the reward function is a key factor for the outcome of the learning process, determining the quality of the learned policy. Many issues can arise due to the choice of the reward signal, such as:

1. *Reward hacking*: the agent may maximize the given reward, without performing the intended task [3, 52].
2. *Reward shaping*: in many applications, the reward does not only define the goal, but it also guides the agent towards a correct solution. Balancing these two components is non-trivial and it is usually handled via a trial-and-error process [38].
3. *Infinite rewards*: in some applications the most suitable choice for a reward signal may include infinite-valued rewards. However, this would violate the assumptions underlying classic RL algorithms, and one must always specify finite values for R .
4. *Multi-objective rewards*: The *Sutton hypothesis* states that *all of what we mean by goals and purposes can be well thought of as the maximization of the cumulative*

sum of a received scalar signal (reward). Whether this statement is true is a heated debate in the AI research community [51, 62]. However, it is clear that in many practical cases, the agent needs to act pursuing different, contrasting objectives: the Multi-Objective RL [MORL, 25] framework tackles these problems, by assuming the reward function \mathbf{R} to be multi-dimensional.

The Preference-based Reinforcement Learning [PbRL, 68] paradigm arose to address these problems using a relative preference-based feedback signal, instead of an absolute, numerical feedback. In this paradigm, the agent selects two trajectories τ_1, τ_2 (generated by following policies π_1 and π_2) to be labeled by a human expert, which returns whether τ_1 is preferred over τ_2 ($\tau_1 \succ \tau_2$), or vice versa ($\tau_1 \prec \tau_2$). It is worth noting that this is not the only type of human feedback found in the literature, in fact Reinforcement Learning from Human Feedback [RLHF, 30] is the branch of RL containing PbRL, and dealing with a myriad of diverse human feedback types (e.g., trajectory critiques, task descriptions, improvements, emergency stops, etc.). [29] provides a practical framework to unify all those different feedback types.

Markov Decision Processes with Preferences A finite-horizon Markov Decision Process with Preferences [MDPP, 68] is a tuple $\langle \mathcal{S}, \mathcal{A}, H, P, \mu_0, \rho \rangle$ where:

- $\langle \mathcal{S}, \mathcal{A}, H, P, \mu_0 \rangle$ is a finite-horizon MDP $\setminus \mathbf{R}$.
- $\rho : \mathcal{T} \times \mathcal{T} \rightarrow [0, 1]$ is the unknown probability function that the expert uses to label pairs of trajectories. In MDPPs, incomparabilities are not modelled, and $\rho(\tau_1, \tau_2) = P(\tau_1 \succ \tau_2)$ denotes the probability that τ_1 is preferred to trajectory τ_2 . A value of $\rho(\tau_1, \tau_2) = 0.5$ denotes complete indifference in choosing a preference between the two trajectories. The following properties must hold for ρ : (a) complementarity, i.e., $\forall \tau_1, \tau_2 \in \mathcal{T} : \rho(\tau_1, \tau_2) = 1 - \rho(\tau_2, \tau_1)$, (b) reflexivity, i.e., $\forall \tau \in \mathcal{T} : \rho(\tau, \tau) = 0.5$.

Moreover, we denote as $\zeta := \{\zeta_i\} = \{\tau_{i1} \succ \tau_{i2}\}_{i \in [N]}$ the set of observed preferences, sampled from ρ . We usually denote by N the cardinality of such set.

Objective and methods The general objective of a PbRL agent is to find a policy π^* that is maximally compliant with the observed preferences ζ , i.e., a policy that maximizes the probability $P_\pi(\tau)$ of generating trajectories that appear as preferred in ζ and minimizes the probability of generating the non-preferred ones in ζ . We can say that a preference $\tau_1 \succ \tau_2 \in \zeta$ is satisfied by policy π if the agent plays τ_1 with more probability with respect to τ_2 :

$$\tau_1 \succ \tau_2 \iff P_\pi(\tau_1) > P_\pi(\tau_2), \quad (2.12)$$

where:

$$P_\pi(\tau) = \mu(s_0) \prod_{i=1}^{|\tau|} \pi(a_i|s_i)P(s_{i+1}|s_i, a_i) \quad (2.13)$$

is the probability of playing τ under policy π .

To learn a policy, each PbRL algorithm in the literature tries to either:

1. *directly learn a policy*, by assuming a parametric policy and maximizing over the maximum a posteriori $P(\pi|\zeta)$ [65], or maintaining a set of candidate policies Π , ranking them using ζ and returning the first policy in the ranking [11, 12]. Recently, many works obtained an optimal policy by expressing a loss over a preference dataset in terms of the parameters of the (parametric) policy. Some notable examples are DPO [47], SLiC-HF [73], Ψ PO [4].
2. *learn a preference model*, i.e., a binary classifier $C(a \succ a'|s)$, trained via supervised learning on ζ , capable of expressing a preference over a pair of actions played in the same state s . Once one has such a classifier, a policy can be obtained by selecting, in each state, the action beating most other actions in that state [24, 66].
3. *learn a utility function* (also commonly called *reward model*) $U : \mathcal{T} \rightarrow \mathbb{R}$ such that:

$$\tau_1 \succ \tau_2 \iff U(\tau_1) > U(\tau_2), \quad (2.14)$$

to then be used as reward function by any RL method after retrieving, in a suitable way, a utility that depends on state-action pairs $U(s, a)$. This is by far the most popular approach in PbRL [2, 16, 49, 54]. If the dataset does not contain incomparable pairs, a scalar utility possessing property in Equation 2.14 always exists [63].

Most works employ a linear utility function, i.e.:

$$U(\tau) = \mathbf{w}^T \cdot \phi(\tau), \quad (2.15)$$

where:

- $\phi(\tau) : \mathcal{T} \rightarrow \mathbb{R}^k$ is a known *trajectory feature mapping*, that needs to be suitably selected by an expert of the field, and is problem-dependent.

- $\mathbf{w} \in \mathbb{R}^k$ is a learnable parameter vector that encodes the weight (importance) of each trajectory feature in obtaining a high utility value for that trajectory.

One should aim at learning \mathbf{w} from preference data, finding a value that minimizes a suitable loss function: intuitively, it should take into account all preferences in the dataset and penalize whenever the property in Equation 2.14 is violated. Generally, the loss function is written as:

$$\mathcal{L}(\mathbf{w}, \zeta) = \sum_{i=1}^{|\zeta|} \alpha_i L(\mathbf{w}, \zeta_i), \quad (2.16)$$

where L is the *pairwise disagreement loss*, and typically $L(\mathbf{w}, \zeta_i) \propto -d(\mathbf{w}, \zeta_i)$, where $d(\mathbf{w}, \zeta_i) := U(\tau_{i1}) - U(\tau_{i2})$ (therefore maximizing the distance in utility between compared trajectories).

The great advantage of using a reward model is that, once a suitable parameter \mathbf{w} is found, one can evaluate new pairs of trajectories without asking the expert for explicit feedback. However, a bias is introduced by the estimation of \mathbf{w} .

The general scheme of PbRL with utility learning is illustrated in Algorithm 2.1. *Note: the trajectory set (\mathcal{T}) generation strategy illustrated at lines 3-9 of Algorithm 2.1 is only one possible choice. See [68], Section 3.4.1 for an exhaustive summary of trajectory generation strategies for PbRL.*

Algorithm 2.1 Utility-based PbRL [68]

Require: Initial policy π_0 , iteration limit m , state sample limit k , rollout limit n

```

1:  $\zeta \leftarrow \emptyset$ 
2: for  $i = 0$  to  $m$  do
3:    $\mathcal{T} \leftarrow \emptyset$ 
4:   for  $j = 0$  to  $k$  do
5:      $s \sim \mu(s)$ 
6:     for  $t = 0$  to  $n$  do
7:        $\mathcal{T} \leftarrow \mathcal{T} \cup \text{ROLLOUT}(s, \pi_i)$  ▷ Generate a trajectory
8:     end for
9:   end for
10:   $(\tau_1, \tau_2) \leftarrow \text{CREATEQUERY}(\mathcal{T})$  ▷ Choose trajectories to evaluate
11:   $\zeta \leftarrow \zeta \cup \text{OBTAINTRAJECTORYPREFERENCES}(\tau_1, \tau_2)$  ▷ Ask the expert
12:   $U^{\pi_i} \leftarrow \text{COMPUTEUTILITYFUNCTION}(\zeta)$  ▷ Reward learning step
13:   $\pi_{i+1} \leftarrow \text{COMPUTEPOLICY}(U^{\pi_i})$  ▷ Calculate policy using DP or RL methods
14: end for
15: return improved policy  $\pi_m$ 

```

Bradley-Terry model Many choices of loss functions for utility learning have been considered in previous works, such as binary [54], hinge [2] and piecewise linear [67] losses. However, most of the work [16, 27, 42] employs a smooth, differentiable sigmoidal-shaped loss. A popular choice is to model the human expert probabilistically using the *Bradley-Terry model* [10]:

$$P(\tau_1 \succ \tau_2) := \sigma(U(\tau_1) - U(\tau_2)) = \frac{1}{1 + \exp(-(U(\tau_1) - U(\tau_2)))}. \quad (2.17)$$

This is equivalent to saying that we treat outcomes $\tau_1 \succ \tau_2$ as independent Bernoulli random variable $Bernoulli(p_{12})$ where the log-odds corresponding to $p_{12} := P(\tau_1 \succ \tau_2)$ is equal to the difference of the utilities:

$$\log \frac{p_{12}}{1 - p_{12}} = U(\tau_1) - U(\tau_2). \quad (2.18)$$

The Bradley-Terry model is usually optimized via Maximum Likelihood Estimation (MLE), by considering the minimum point of the negative log-likelihood function:

$$\hat{\mathbf{w}}^{MLE} = \arg \min_{\mathbf{w}} -\log(P(\zeta|\mathbf{w})) = \arg \min_{\mathbf{w}} \left\{ -\sum_{i=1}^{|\zeta|} \log \left(\frac{1}{1 + \exp(-(U(\tau_{i1}) - U(\tau_{i2})))} \right) \right\}. \quad (2.19)$$

Note: in Equation 2.19, we consider a dataset structured such that all samples belong to the \succ class.

Or, alternatively, via a Bayesian approach, using Maximum A Posteriori (MAP), a method that allows to specify a custom prior distribution for the parameter vector, i.e., $P(\mathbf{w})$:

$$\hat{\mathbf{w}}^{MAP} = \arg \max_{\mathbf{w}} P(\zeta|\mathbf{w})P(\mathbf{w}). \quad (2.20)$$

It can be proved that the log-likelihood is convex in \mathbf{w} when the argument of the sigmoid is linear with respect to \mathbf{w} , that is, when the utility function is linear in \mathbf{w} . Therefore, standard methods of *convex optimization* [9] can be used to obtain an optimal point, and the set of optimum points is convex.

The model is invariant to translations, in fact, if all the utilities $U(\tau_i)$, $i = 1, \dots, |\mathcal{T}|$ are

scaled by an additive constant $c \in \mathbb{R}$ the probability outputs stay the same since:

$$P(\tau_i \succ \tau_j) = \sigma(U(\tau_i) - U(\tau_j)) = \sigma((U(\tau_i) + c) - (U(\tau_j) + c)). \quad (2.21)$$

The use of this preference model is motivated by several reasons: firstly, the differentiability and convexity of the resulting negative log-likelihood function make it computationally fast to optimize; then, this model complies with Luce’s axiom of choice [45], which requires that the probability of choosing a specific option from a pool does not depend on the presence or absence of other options in the pool; moreover [29] prove that it is the maximum entropy distribution over choices for an expert who is making in expectation a choice with an optimal reward, up to some slack $\epsilon > 0$.

In the RLHF literature, some works explore the possibility for the expert to choose the preferred trajectory from a pool $\{\tau_1, \tau_2, \dots, \tau_N\}$: in this case the probabilistic model that is consistent with Luce’s axiom of choice and that extends the Bradley-Terry one is the *multinomial logit* model [57]:

$$P(\tau_i \text{ is chosen as preferred}) = \frac{\exp(U(\tau_i))}{\sum_{j=1}^N \exp(U(\tau_j))}. \quad (2.22)$$

If the feedback of the expert consists of a full ranking (total order) from a pool of $N > 2$ trajectories, the Plackett-Luce [45] model is the natural extension complying with Luce’s axiom of choice:

$$P(\tau_1 \succ \tau_2 \succ \dots \succ \tau_N) = \frac{p_1}{\sum_{i=1}^N p_i} \frac{p_2}{\sum_{i=2}^N p_i} \dots \frac{p_{N-1}}{p_{N-1} + p_N}, \quad (2.23)$$

where $p_i := \exp(U(\tau_i))$.

In other applications, if the set of items for the expert to choose from is continuous, the *Boltzmann distribution* [30] is used.

It is worth noticing that the Bradley-Terry model has also been applied in a variety of other applications, from sports to chess Elo system [21], to comparisons of journals [53] and LLMs [15].

2.6. Multi-Objective Reinforcement Learning

In the real world, agents often need to tackle sequential decision-making problems in which many conflicting objectives need to be pursued. For example, in an autonomous

driving setting, the agent may want to minimize travel time, while ensuring safety and comfort during the trip, also minimizing the cost and CO2 emissions may well be desirable objectives.

One could simply approach such problems by combining the different objectives into an additive scalar reward, inserting penalties for negative events and rewards for positive events. However, this is typically a "blind" approach in which the engineer needs to iteratively try different scalarizations of the objectives, until a good performance is achieved, but not all choices of scalarization are explored and such scalarization weights may change over time, requiring model re-training. Moreover, this approach reduces explainability of the result, since all the objectives are aggregated and we cannot inspect which objectives are maximized by some policy.

A Markovian environment with a multi-objective reward is formalized by the following model:

Multi-Objective Markov Decision Process (MOMDP) A finite-horizon, multi-objective MDP [MOMDP, 25] is a tuple $\mathcal{M} := \langle \mathcal{S}, \mathcal{A}, H, P, \mathbf{R}, \mu_0 \rangle$ where:

- $\langle \mathcal{S}, \mathcal{A}, H, P, \mu_0 \rangle$ is a finite-horizon MDP $\setminus \mathbb{R}$.
- \mathbf{R} is the d -dimensional reward function $\mathbf{R} : \mathcal{S} \times \mathcal{A} \times \llbracket H \rrbracket \rightarrow \mathbb{R}^d$, $\mathbf{R}(s, a, t)$ specifies the reward that the agent receives from the environment by playing action a in state s at timestep t .

Given any scalarization function $u : \mathbb{R}^d \rightarrow \mathbb{R}$, a scalar reward MDP can be obtained as $\langle \mathcal{S}, \mathcal{A}, H, P, u(\mathbf{R}), \mu_0 \rangle$.

Similarly to the scalar case, the value function is defined as the multi-dimensional expected reward following a certain policy π starting from state s at timestep t :

$$\mathbf{V}^\pi(s) = \mathbb{E} \left[\sum_{k=t+1}^H \mathbf{r}_k | \pi, s_t = s \right] \in \mathbb{R}^d, \quad (2.24)$$

$$\mathbf{V}^\pi = \mathbb{E} \left[\sum_{k=0}^H \mathbf{r}_k | \pi, \mu_0 \right] \in \mathbb{R}^d. \quad (2.25)$$

Clearly, we can have policies π_1 and π_2 such that $V_i^{\pi_1} > V_i^{\pi_2}$ and $V_j^{\pi_1} < V_j^{\pi_2}$ for $i \neq j$, and therefore neither policy π_1 nor π_2 dominates the other. For this reason, usually no single optimal value vector exists, and one should search for peculiar subsets of the set Π

of all admissible Markovian stochastic policies, such as:

1. The Pareto Front (PF), i.e., the non-dominated set of policies:

$$PF(\Pi) = \{\pi \in \Pi \mid \nexists \pi' \in \Pi : \mathbf{V}^{\pi'} \succ \mathbf{V}^\pi\}, \quad (2.26)$$

where \succ is the Pareto dominance pre-order, i.e., $\mathbf{V}^{\pi_1} \succ \mathbf{V}^{\pi_2} \iff (\forall i : V_i^{\pi_1} \geq V_i^{\pi_2}) \wedge (\exists i : V_i^{\pi_1} > V_i^{\pi_2})$ (the same pre-order translates to a pre-order on corresponding policies).

2. The Convex Hull (CH) is the set of all policies that are optimal for some choice of linear scalarization $u(\mathbf{V}^\pi) = \mathbf{w}^T \mathbf{V}^\pi$:

$$CH(\Pi) = \{\pi \in \Pi \mid \exists \mathbf{w}, \forall \pi' \in \Pi : \mathbf{w}^T \mathbf{V}^\pi \geq \mathbf{w}^T \mathbf{V}^{\pi'}\}. \quad (2.27)$$

The majority of methods to solve MOMDPs take either an *a priori* approach, where the user needs to specify beforehand a scalarization function, and whose output is a single optimal policy or an *a posteriori* approach in which a policy set of interest is retrieved and presented to the user for selection (a notable example is Convex Hull Value Iteration [CHVI, 5]). A third category is the *interactive* approach, allowing the user to specify its preferences iteratively during the learning process [60, 61].

3 | Related Works

In this chapter, we present the main works that lay the foundations for developing our contributions. In particular, in Section 3.1 we present selected works from the PbRL and RLHF literature that consider, in some way, the possibility for the expert labeler to mark a pair of trajectories as *indifferent* or *incomparable*. Then, in Section 3.2 we introduce a recent work that, for the first time, approaches the problem of representing (pre-)order relations with numerical, multi-dimensional utilities from a theoretical perspective, analyzing what are the computational limits of these operations.

3.1. PbRL Methodologies for Indifference and Incomparability

The vast majority of the literature in PbRL ignores the fact that a human labeler could not be able to indicate a strict preference between a pair of trajectories. This means that, given (τ_1, τ_2) , the only considered comparison outcome is often either $\tau_1 \succ \tau_2$ or $\tau_1 \prec \tau_2$, frequently disregarding the possibility for the two trajectories to be indifferent to each other (i.e., $\tau_1 \asymp \tau_2$), or, even more frequently, to be incomparable to each other (i.e., $\tau_1 \parallel \tau_2$).

This issue was first noted and discussed by [68], including it among future challenges in PbRL, pointing out that no algorithm was able to generate a set of Pareto-optimal policies in the case of incomparabilities in the dataset. Moreover, the authors present a research direction aimed at solving this particular problem: to use a reward modeling approach in which the learned utility is non-scalar, allowing it to differ in multiple dimensions, and then to use MORL on the induced MOMDP to learn a set of Pareto-optimal policies. Surprisingly, even considering the wide success of this survey paper, to the best of our knowledge, no work has ever pursued this proposed approach.

Attempts have been made to deal with indifferences in the dataset. One example is [12], in which the authors assume that, in the case of equally preferred trajectories, both receive

the same preference probability by the preference model, i.e.:

$$\tau_1 \asymp \tau_2 \in \zeta \iff \rho(\tau_1 \prec \tau_2) = \rho(\tau_1 \succ \tau_2) = 0.5. \quad (3.1)$$

This idea has then been frequently utilized, and has been, from then on, the most popular beyond ignoring indifferences in the dataset.

[16] and [27] parametrize a reward model using a deep neural network, obtaining a highly non-linear utility function, and their approach is to equally weight indifferences as in Equation 3.1 and to discard incomparabilities. The choice of ignoring incomparabilities is nonetheless made blindly, implicitly making the assumption that little information is carried in such examples. This choice prevents the definition of a multi-dimensional problem, since, as already discussed, without incomparabilities in the dataset, a scalar utility is sufficient to represent the pre-order relation.

Albeit PbRL existing for more than a decade, only recently has MORL with preference feedback gained the attention of the scientific community. [37] trains a multi-dimensional reward model from trajectory preferences. However, the framework presented in the paper requires the presence of an oracle $P(\tau_1 \succ \tau_2 | \mathbf{w})$ able to classify a pair with respect to an (arbitrary) scalarization of multiple objectives. Although this might be a possibility in some real-life scenarios, it assumes the oracle to precisely know the dimensions of the reward model, which, in most of practical cases, is false. As an example, if the same query is presented to multiple users who are unaware of the model dimensions, each will provide a feedback with respect to some objectives which may differ between users.

Extensions of the Bradley–Terry model that accommodate ties have been proposed [17, 48]. These models were created to be applied to problems of ranking individual and distinct items (such as chess players or sports teams) and were almost never utilized in PbRL. Specifically, the Bradley-Terry with Ties model [48] introduces a "threshold" parameter η , to be jointly estimated with the utilities, in the following way:

$$\begin{aligned} P(i \succ j) &= \sigma(U_i - U_j - \eta), \\ P(i \prec j) &= \sigma(U_j - U_i - \eta), \\ P(i \asymp j) &= 1 - \sigma(U_i - U_j - \eta) - \sigma(U_j - U_i - \eta). \end{aligned} \quad (3.2)$$

It can be seen that, for $\eta = 0$, this model is equivalent to the Bradley-Terry model. The higher η , and the more the utility values U_i and U_j are similar, the more the model will lean towards a tie.

A similar model is formulated by [17]. The author critiques the aforementioned extension by debating that such a model does not comply with Luce’s axiom of choice [45], and proposes a different model:

$$\begin{aligned}
 P(i \succ j) &= \frac{e^{U_i}}{e^{U_i} + e^{U_j} + \nu e^{(U_i+U_j)/2}}, \\
 P(i \prec j) &= \frac{e^{U_j}}{e^{U_i} + e^{U_j} + \nu e^{(U_i+U_j)/2}}, \\
 P(i \asymp j) &= \frac{\nu e^{(U_i+U_j)/2}}{e^{U_i} + e^{U_j} + \nu e^{(U_i+U_j)/2}}.
 \end{aligned} \tag{3.3}$$

Where ν is again a parameter to be estimated controlling the ties probability (if $\nu = 0$, this model is equivalent to the Bradley-Terry one). In this case, the tie probability is proportional to the geometric mean of $\ln(U_i)$ and $\ln(U_j)$.

Recently, few works [35, 36] used model in Equation 3.2 in reward modeling, using a dataset of labeled preferences and indifferences between pair of trajectories. In [36], the authors claim that the inclusion of ties in reward modeling permits to learn utilities such that the distributions of $\Delta U^i := U(\tau_{i1}) - U(\tau_{i2})$ for indifferent and preferred pairs are substantially different, allowing the model to clearly distinguish between strict preferences and indifferences based on ΔU . However, this was only demonstrated through experiments and further studies are necessary to understand the theoretical advantages of using such BT extensions in PbRL.

3.2. Computational Issues in Multi-Objective PbRL

As already discussed in Section 2.5, PbRL methodologies aim at learning a policy either directly from human preference feedback, from a surrogate preference model or a surrogate reward model, which can take as argument a trajectory ($U : \mathcal{T} \rightarrow \mathbb{R}$) or a state-action pair, obtaining a Markovian reward model $R : \mathcal{S} \times \mathcal{A} \rightarrow \mathbb{R}$. Directly learning a policy is the most unbiased and general approach, since no numerical signal is ever required and the learned policy should be able to inherently represent incomparabilities. However, from a computational point of view, assessing policy dominance (and therefore optimality) is conjectured to be intractable.

This aspect is explored in depth in [19], a work that analyzes the computational challenges in learning from preference feedback by only assuming the presence of a (pre-)order relation as a dataset (and without assuming a probabilistic model ρ for the expert or an underlying "true" reward function). In the following, we restate the main contributions

of [19], that lay the foundations of this work and that we will refer to in Chapter 4.

Definition 3.1 (Compatible Utility - Partial Preorder). *Let $\preceq_{\mathcal{T}}$ be a preorder over \mathcal{T} and let $u : \mathcal{T} \rightarrow R^m$ with $m \in \mathbb{N}$ be a multi-dimensional utility. u is compatible with $\preceq_{\mathcal{T}}$ if for every $\tau, \tau' \in \mathcal{T}$ it holds that $\tau \preceq_{\mathcal{T}} \tau' \iff \mathbf{u}(\tau) \preceq \mathbf{u}(\tau')$, where \preceq is the component-wise order of the utility vectors. Precisely, we have:*

- $\tau_1 \succ \tau_2 \implies \forall i \in \llbracket m \rrbracket : u_i(\tau_1) \geq u_i(\tau_2) \wedge \exists j \in \llbracket m \rrbracket : u_j(\tau_1) > u_j(\tau_2)$,
- $\tau_1 \asymp \tau_2 \implies \forall i \in \llbracket m \rrbracket : u_i(\tau_1) = u_i(\tau_2)$,
- $\tau_1 \parallel \tau_2 \implies \exists i, j \in \llbracket m \rrbracket : i \neq j \wedge u_i(\tau_1) > u_i(\tau_2) \wedge u_j(\tau_1) < u_j(\tau_2)$.

If the preorder is total, the case is much simpler: a one-dimensional utility that is compatible with the preorder can always be constructed by sorting elements with respect to $\preceq_{\mathcal{T}}$ and assigning a non-decreasing utility value to elements according to this order. In the case of a partial pre-order, obtaining a compatible utility is not straightforward and it can be proved that (a) the minimum dimensionality of such utility is equal to the order dimension of the preorder and that (b) the construction of a minimal dimensionality utility is NP-hard [19, Theorem 4.2], a result that follows from the NP-hardness of computing the order dimension.

Through the following example, the authors prove that the Pareto optimality of a policy π with respect to a compatible utility \mathbf{u} does not imply Pareto optimality of π with respect to another compatible utility \mathbf{u}' :

Example 3.1. *Let $\mathcal{T} = \{\tau_1, \tau_2, \tau_3\}$, and the total order $\leq_{\mathcal{T}}$ be defined as:*

$$\tau_1 >_{\mathcal{T}} \tau_2 >_{\mathcal{T}} \tau_3. \quad (3.4)$$

Let $\Pi = \{\pi, \pi'\}$ be the policy space with the corresponding trajectory distributions $d_{\pi} = (0.5, 0.5, 0)^{\top}$ and $d_{\pi'} = (0.8, 0, 0.2)^{\top}$; Let the utility values for the three trajectories be $u_1 = (4, 2, 0)^{\top}$ and $u_2 = (4, 2, -2)^{\top}$ (both compatible with $\leq_{\mathcal{T}}$). Let $J(\pi; u) = \mathbb{E}_{\tau \sim \pi}[u(\tau)]$ We have:

$$\begin{aligned} J(\pi; u_1) &= J(\pi; u_2) = 3, \\ J(\pi'; u_1) &= 3.2, \quad J(\pi'; u_2) = 2.8. \end{aligned} \quad (3.5)$$

Therefore, π' u_1 -Pareto dominates π and π u_2 -Pareto dominates π' .

This can also be seen for partial (pre-)orders. Due to this result, a definition of dominance

that is independent of the choice of the compatible utility function must be used. To this end, the authors define the following dominance relation.

Definition 3.2 (Policy Dominance for Partial Orders). *Let $\preceq_{\mathcal{T}}$ be an order over \mathcal{T} , and let $\pi, \pi' \in \Pi$ be two policies. $\pi \preceq_{\mathcal{T}}$ -strictly dominates π' if, for every utility $\mathbf{u} : \mathcal{T} \rightarrow \mathbb{R}$ compatible with $\preceq_{\mathcal{T}}$, it holds that:*

$$\mathbf{J}(\pi; \mathbf{u}) - \mathbf{J}(\pi'; \mathbf{u}) = \langle d_{\pi} - d_{\pi'}, \mathbf{u} \rangle \succ \mathbf{0}. \quad (3.6)$$

Clearly, checking this condition for *every* compatible utility \mathbf{u} is infeasible. Through Theorem 5.2 of [19], this condition is proved to be equivalent to stating that π strictly dominates π' if and only if, for every possible realizer $\{\leq_{\mathcal{T},i}\}_{i \in \llbracket m \rrbracket}$ of $\preceq_{\mathcal{T}}$, it holds that:

$$\forall i \in \llbracket m \rrbracket : \pi' \leq_{\Pi,i} \pi \wedge \exists j \in \llbracket m \rrbracket : \pi' <_{\Pi,j} \pi. \quad (3.7)$$

Therefore requiring, equivalently, to check dominance between policies with respect to all total orders that are linear extensions of $\preceq_{\mathcal{T}}$. Even if checking this condition for a single total order is as simple as computing a cumulative sum of probabilities of playing trajectories ordered by $\leq_{\mathcal{T},i}$ (which is computable in $O(|\mathcal{T}|)$), this operation needs to be repeated for all linear extensions of $\preceq_{\mathcal{T}}$, i.e., in the worst case, $|\mathcal{T}|!$ times.

The authors conjecture the problem of assessing dominance between policies with respect to partial orders to be indeed computationally hard. Since the notion of optimality of a policy in Π is based on Pareto dominance, even the problem of checking optimality for a policy may thus be unsolvable in polynomial time.

4 | Problem Setting and Reward Model Desiderata

As introduced in Chapter 1, the objective of this work is to explore the overlooked (as presented in Chapter 3) setting of having incomparabilities in the dataset and its connection to multi-objective sequential decision-making problems. In particular, we want to address the theoretical issues in defining a suitable multi-dimensional reward learning method from this type of feedback, and then proposing some reasonable model choices addressing this problem and facing these issues.

This chapter is organized as the following. In Section 4.1, we introduce the motivations for developing our contributions, relying on the computational negative results presented in Section 3.2. Then, in Section 4.2, we formalize the problem we deal with and the semantic desiderata of a PbRL agent’s preference model acting on such problem. Finally, in Section 4.3, we present an important negative result concerning all models complying with our desiderata for a reward model.

4.1. Why the Need for Multi-Dimensional Reward Models

Multi-dimensional reward models are motivated by both theoretical results and practical necessities. From a theoretical perspective, as stated in Section 3.2, [19] conjectures the computational hardness of assessing policy optimality in the case of a partial pre-order of trajectories $\preceq_{\mathcal{T}}$, i.e., a dataset that contains incomparable pairs of trajectories. This computational barrier motivates the need to shift towards utilizing numerical utilities and rewards that allow for more convenient computational properties, at the cost of introducing a bias (because we need to encode quantitatively how much a trajectory is, for example, preferred with respect to another, while the feedback is a simple binary comparison).

Moreover, the need of a reward model approach that can model preference relations that

are not total orders is strengthened by practical needs: in real cases, the expert might not be able to indicate a strict preference and would like to mark a pair of trajectories as incomparable. This can be due to different reasons:

1. The trajectories may be contrasting with respect to multiple objectives, that are evaluated as a contrasting trade-off by the expert when labeling the pair but such objectives are not (completely) known to the designer of the learning algorithm.
2. The objectives may be implicit or unknown to the expert, who can nevertheless solve the task but is unable to express the underlying reasoning - a scenario that is common in machine learning applications.
3. The trajectories taken into consideration are not comparable for a different, intrinsic reason. For example, in chess matches, if two trajectories start from different initial states, they solve problems that may be far apart from each other in terms of complexity and needed reasoning or problem-solving skills.

On the other hand, *indifference* is an orthogonal issue: semantically, it means that the expert was not able to express a strict preference due to the fact that the trajectories are very similar in terms of underlying objectives (which may be known to him up to a certain degree). A principled approach should be able to account for both possibilities, allowing for a partial pre-order dataset ζ of trajectories.

Finally, to the best of our knowledge, as previously noted in Section 2.5, no work has ever dealt with this specific setting, and, in particular, the possibility of incomparabilities in the dataset and the connection to multi-dimensional reward models has been completely overlooked so far.

4.2. Problem Setting and Model Desiderata

In what follows, we first describe, in Section 4.2.1, the particular setting of the problem, formalizing it using an extension of the MDPP framework. Subsequently, in Section 4.2.2, we develop a set of desirable conditions that a reward model should satisfy under this setting.

4.2.1. Problem Setting

We will make use of a slightly modified version of finite-horizon MDPP (see Section 2.5) in modeling the problem environment. We will consider, differently from [68] and all previous works in PbRL, an expert capable of, given a trajectory pair $\{\tau_1, \tau_2\}$, providing

feedback belonging to one of four distinct classes, namely:

- $\tau_1 \succ \tau_2$, i.e., τ_1 is *strictly preferred* to τ_2 .
- $\tau_1 \prec \tau_2$, i.e., τ_2 is strictly preferred to τ_1 .
- $\tau_1 \asymp \tau_2$, i.e., the expert is *indifferent* when it comes to choosing between τ_1 and τ_2 .
- $\tau_1 \parallel \tau_2$, i.e., the expert evaluates the two trajectories to be *incomparable*.

We choose to model the expert probabilistically, following the PbRL literature, and therefore define the unknown probability function that the expert uses to label pairs of trajectories as $\rho : \mathcal{T} \times \mathcal{T} \rightarrow \Delta(\mathcal{Y})$, where $\mathcal{Y} := \{\succ, \prec, \asymp, \parallel\}$, such that:

$$\rho(\tau_1, \tau_2) = (\rho_{\succ}, \rho_{\prec}, \rho_{\asymp}, \rho_{\parallel})^\top = (P(\tau_1 \succ \tau_2), P(\tau_1 \prec \tau_2), P(\tau_1 \asymp \tau_2), P(\tau_1 \parallel \tau_2))^\top. \quad (4.1)$$

ρ must possess the *reflexivity* property: $\forall \tau \in \mathcal{T} : \rho(\tau, \tau) = (0, 0, 1, 0)^\top$, i.e., when two equal trajectories are compared, the expert assigns all probability mass to the "indifference" class. This is intuitively motivated by the fact that the compared trajectories are equal and therefore the expert is completely indifferent in choosing one or the other.

We will also consider the presence of a dataset of collected feedback from the expert $\zeta = \{(\tau_{i1}, \tau_{i2}, y_i)\}_{i \in [N]}$, where each label y_i , assigned to the trajectory pair $\{\tau_{i1}, \tau_{i2}\}$, is sampled according to ρ , i.e., $y_i \sim \rho(\tau_{i1}, \tau_{i2})$.

An alternative interpretation for ζ is to consider a relation $R_\zeta \in \mathcal{T} \times \mathcal{T}$ induced by the collected preferences and indifferences in ζ . Such relation is constructed such that the following property holds:

$$\forall \tau_1, \tau_2 \in \mathcal{T} : (\tau_1, \tau_2) \in R_\zeta \iff \{\tau_1, \tau_2, \succ\} \in \zeta \vee \{\tau_2, \tau_1, \prec\} \in \zeta \vee \{\tau_1, \tau_2, \asymp\} \in \zeta \vee \{\tau_2, \tau_1, \asymp\} \in \zeta. \quad (4.2)$$

It can be trivially verified that there exist instances of datasets such that the relation R_ζ is neither reflexive nor transitive. Therefore, differently from the Preference-based MDP formalism introduced by [19, Section 3], the relation corresponding to our dataset of trajectory comparisons is not a pre-order.

4.2.2. Desiderata for a Reward Model

As discussed in Section 2.5, the most popular approach in designing PbRL algorithms is the use of a reward model, to be used as reward function of a suitably chosen MDP by

any linear programming, planning-based (DP) or learning-based (RL) method of choice. This is the approach that will be pursued in our work.

The most novel feature of our problem setting is the possibility for the expert to label trajectory pairs as incomparable. Theoretical works have proved the existence of a multi-dimensional numerical utility that represents a partial pre-order relation [22, 41]. The connection between multi-dimensionality of the reward function and incomparabilities is also covered in MORL [25] and PbRL [68] literature.

In a multi-objective setting, if two trajectories are to be compared, the natural choice - the one a human expert would make - is to evaluate the two trajectories with respect to each single objective individually, and then to reason on this intermediate evaluation to output a label for the pair. In mathematical terms, we consider, at least for now, the presence of a true, multi-dimensional, utility function for trajectories $\hat{U} : \mathcal{T} \rightarrow \mathbb{R}^{\hat{d}}$ that evaluates trajectories according to \hat{d} "true" objectives.

The general scheme of the expert's reasoning then becomes:

1. Given the trajectory pair $\{\tau_1, \tau_2\}$, obtain the \hat{d} -dimensional utility scores for τ_1 and τ_2 , i.e., $\hat{U}(\tau_1), \hat{U}(\tau_2) \in \mathbb{R}^{\hat{d}}$.
2. Evaluate the difference between the utility scores $\Delta\hat{U}(\tau_1, \tau_2) := \hat{U}(\tau_1) - \hat{U}(\tau_2)$.
3. Output a label for the comparison $y \in \mathcal{Y}$ that depends on $\Delta\hat{U}(\tau_1, \tau_2)$.

In our specific setting, since we've postulated a probabilistic model for the expert (see Equation 4.1), the label y is randomly sampled from a 4-categorical distribution ρ . Therefore, the mapping from $\Delta\hat{U}(\tau_1, \tau_2)$ to y is non-deterministic. We assume that there exists a suitable $\gamma : \mathbb{R}^{\hat{d}} \rightarrow \Delta(\mathcal{Y})$ such that:

$$y \sim \rho(\tau_1, \tau_2) = \gamma(\Delta\hat{U}(\tau_1, \tau_2)). \quad (4.3)$$

Of course, we do not know ρ , and neither γ . It is worth to note that this is only a modelization of the expert, and that, in the real world, the expert could output a label following a procedure that is very different from the one we presented. On top of this assumption, another choice needs to be made: the modelization of γ . This modelization is an arbitrary choice, similarly to the customary choice of the Bradley-Terry model or the Plackett-Luce model in the literature, and, as such, there are no real guarantees that the expert would comply with such a model. Assuming that feedbacks are generated according to Equation 4.3, we can postulate a set of desired properties for γ :

Definition 4.1 (Rational Multi-Objective Expert Model). *A probability function $\gamma : \mathbb{R}^d \rightarrow \Delta^3$ is said to be a rational multi-objective expert model if and only if the following conditions are satisfied:*

1. $\gamma(\Delta\mathbf{U}) \rightarrow (1, 0, 0, 0)^\top$ as $\Delta\mathbf{U} \rightarrow (+\infty, \dots, +\infty)^\top$.
2. $\gamma(\Delta\mathbf{U}) \rightarrow (0, 1, 0, 0)^\top$ as $\Delta\mathbf{U} \rightarrow (-\infty, \dots, -\infty)^\top$.
3. $\gamma(\Delta\mathbf{U}) \rightarrow (0, 0, 1, 0)^\top$ as $\Delta\mathbf{U} \rightarrow \mathbf{0}_d$.
4. $\forall \mathbf{v} \in \{-1, 1\}^d$ s.t. $\mathbf{v} \neq \pm \mathbf{1}_d$: $\gamma(\Delta\mathbf{U}) \rightarrow (0, 0, 0, 1)^\top$ as $\Delta\mathbf{U} \rightarrow \mathbf{v} \odot (+\infty, \dots, +\infty)^\top$.
5. $\forall t \in \mathbb{R} : \gamma_A(t \cdot \mathbf{1}_d) = 0$.
6. Let $\mathcal{V}_\succ := \{(+\infty, \dots, +\infty)^\top\}$, $\mathcal{V}_\prec := \{(-\infty, \dots, -\infty)^\top\}$, $\mathcal{V}_\simeq := \{\mathbf{0}_d\}$, $\mathcal{V}_\parallel := \{\mathbf{v} \odot (+\infty, \dots, +\infty)^\top, \mathbf{v} \in \{-1, 1\}^d$ s.t. $\mathbf{v} \neq \pm \mathbf{1}_d\}$. For all possible choice of vectors \mathbf{x} , \mathbf{y} chosen from distinct sets between \mathcal{V}_\succ , \mathcal{V}_\prec , \mathcal{V}_\simeq and \mathcal{V}_\parallel , the value of γ for the non-zero components are monotone on the line segment joining \mathbf{x} and \mathbf{y} , i.e., on $\{\lambda \mathbf{x} + (1 - \lambda) \mathbf{y}, \lambda \in [0, 1]\}$.

We now provide an intuitive explanation for the conditions above. If the utility difference vector $\Delta\mathbf{U}$ is positive in all the d components, it means that $\mathbf{U}(\tau_1) \succ \mathbf{U}(\tau_2)$, and that, therefore, τ_1 is better than τ_2 along all objectives. For this reason, the expert should lean towards a preference (\succ). The extreme case is a positive unbounded value for each component of $\Delta\mathbf{U}$: in this case the expert should allocate all probability mass in class \succ (Condition 1). An analogous reasoning can be done for Condition 2, by considering the \prec class.

The expert leans towards indifference when $\mathbf{U}(\tau_1) \simeq \mathbf{U}(\tau_2)$, i.e., when $\forall i \in \llbracket d \rrbracket : U_i(\tau_1) \simeq U_i(\tau_2)$. In other words, this happens when $\Delta\mathbf{U}$ is close to the origin in the d -dimensional space. Condition 3 expresses this reasoning, with the addition that, for values of $\Delta\mathbf{U}$ that tend to the origin, the probability mass is shifted completely towards the indifference class (\simeq).

Condition 5 is of fundamental importance since it allows the model γ to be brought back to the scalar utility case (therefore the standard PbRL setting, in which incomparabilities are not considered): in fact, when $\Delta\mathbf{U} = (t, \dots, t)^\top \in \mathbb{R}^d$, the case could be modeled equally using a scalar utility difference ΔU , therefore, on this line ($t \cdot \mathbf{1}_d, t \in \mathbb{R}$), it is suitable to have a probability value for the incomparability (\parallel) class equal to 0. A maximum probability mass should be given to incomparability if some components of $\Delta\mathbf{U}$ have large magnitudes and opposite sign, i.e., if the two trajectories have significantly contrasting objective scores. Condition 5 expresses this property in the case of unbounded

values of ΔU .

Condition 6 expresses the fact that γ should not "oscillate" between the notable points in \mathbb{R}^d mentioned in Conditions 1-5; instead, γ should have a monotone behavior on the line segment connecting each pair of extreme points related to different classes, limited to the components γ_i and γ_j corresponding to such classes.

4.3. Non-Convexity of Negative Log-Likelihood

After selecting a suitable expert model of the form specified in Equation 4.3, one should opt for defining a family of reward models $U(\tau; \mathbf{W})$ parametrized by a parameter \mathbf{W} , and then choosing the value of \mathbf{W} such that the probability of generating the dataset ζ is maximized (this is an instance of *supervised learning*). In mathematical terms, we aim to find the maximum likelihood estimate of \mathbf{W} , i.e.:

$$\hat{\mathbf{W}}^{MLE} = \arg \max_{\mathbf{W}} \prod_{i=1}^N \gamma_{y_i}(\Delta U(\tau_{i1}, \tau_{i2}; \mathbf{W})). \quad (4.4)$$

The maximization of the likelihood is usually replaced by the minimization of the negative log-likelihood function, due to numerical issues in evaluating the likelihood:

$$\hat{\mathbf{W}}^{MLE} = \arg \min_{\mathbf{W}} - \sum_{i=1}^N \log(\gamma_{y_i}(\Delta U(\tau_{i1}, \tau_{i2}; \mathbf{W}))). \quad (4.5)$$

In this section, we present an important negative result regarding the complexity of the optimization procedure of any expert model γ complying with the aforementioned properties.

Theorem 4.1. *There exists no function $f : \mathbb{R} \rightarrow [0, 1]$ such that $\lim_{x \rightarrow -\infty} f(x) = 1 \wedge \lim_{x \rightarrow +\infty} f(x) = 1 \wedge \lim_{x \rightarrow 0} f(x) = 0$ and such that the function $g : [0, 1] \rightarrow \mathbb{R}$ defined as $g(x) = -\log(f(x))$ is convex in x .*

Proof. Trivially, we have:

$$\begin{aligned} \lim_{x \rightarrow -\infty} f(x) = 1 &\implies \lim_{x \rightarrow -\infty} g(x) = 0, \\ \lim_{x \rightarrow +\infty} f(x) = 1 &\implies \lim_{x \rightarrow +\infty} g(x) = 0, \\ \lim_{x \rightarrow 0} f(x) = 0 &\implies \lim_{x \rightarrow 0} g(x) = +\infty. \end{aligned} \quad (4.6)$$

Let $\bar{x} \in \mathbb{R}^+$, $\lambda = \frac{1}{2}$. Then:

$$\begin{aligned} \lambda g(-\bar{x}) + (1 - \lambda)g(\bar{x}) &= \frac{1}{2}g(-\bar{x}) + \frac{1}{2}g(\bar{x}) = c \\ &< +\infty = g(0) = g\left(\frac{1}{2}(-\bar{x}) + \frac{1}{2}\bar{x}\right) = g(\lambda(-\bar{x}) + (1 - \lambda)\bar{x}), \end{aligned} \quad (4.7)$$

for a suitable $c \in \mathbb{R}$. Therefore, since inequality of Equation 2.2 is violated, the function g is not convex in its argument x . \square

Remark 4.1 (Non-Convexity of Negative Log-Likelihood). Let $\mathbf{v} \in \{-1, 1\}^d$ s.t. $\mathbf{v} \neq \pm \mathbf{1}_d$. We note that f is the restriction on $t \cdot \mathbf{v}$, $t \in \mathbb{R}$ of the incomparability component of a multi-objective expert model $\gamma : \mathbb{R}^d \rightarrow \Delta^3$ that is compliant with Conditions 4 and 5 of Definition 4.1. Moreover, we note that g is the restriction on $t \cdot \mathbf{v}$, $t \in \mathbb{R}$ of the negative log-likelihood function for the incomparability class, i.e.:

$$\begin{aligned} l(\Delta \mathbf{U}) &:= -\log(\gamma_{\parallel}(\Delta \mathbf{U})), \\ g(t) &= l(t \cdot \mathbf{v}). \end{aligned} \quad (4.8)$$

Therefore, $l(\Delta \mathbf{U})$ is not convex in $\Delta \mathbf{U}$. Since $\Delta \mathbf{U}$, i.e., the reward model, is a function of some optimization parameter \mathbf{W} , in the general case, l is not convex in \mathbf{W} . It follows from this result that there exists no model γ that is optimizable via MLE with convex optimization algorithms (differently, for example, from the Bradley-Terry model) and compliant with Definition 4.1.

This fact implies that optimizing such models is a *non-convex optimization* [28] problem, and, as such, it does not exhibit the desirable property of presenting a single minimum (a global one), that is true for convex functions. Instead, a non-convex function may present several (possibly infinite) local minima, to which the optimization algorithm may converge and therefore yield a sub-optimal solution.

It is worth to note that, as demonstrated in the experiments (Chapter 6), in practice, robust optimizers find, even for models in which the loss function is not convex, the optimal parameter estimate with high probability.

5 | Models' Derivations

In this section, we illustrate some possible choices of the expert model γ complying - mostly or entirely - with the definition of rational multi-objective expert model. Since a trade-off between guarantees on the parameter optimization procedure and expressiveness of the model exists, we explain, for each model, the design choices made in light of this trade-off. We begin, in Section 5.1, by presenting the general structure of these models, then, in Sections 5.1.1, 5.1.2 and 5.1.3, we derive and comment the proposed models.

5.1. Extending the Bradley-Terry Model

In the following, we will consider the presence of a suitable trajectory feature mapping $\phi : \mathcal{T} \rightarrow \mathbb{R}^k$. Moreover, we will study the particular case in which the reward model is linear in the trajectory feature vector, i.e., $\mathbf{U}(\tau) = \mathbf{W} \cdot \phi(\tau)$, $\mathbf{W} \in \mathbb{R}^{d \times k}$ (see Equation 2.15 for the one-dimensional case), but a non-linear utility extension is nonetheless possible and its usage is straightforward.

In choosing a suitable γ such that it is a rational multi-objective expert model, we focus on designing a model that extends the Bradley-Terry model (see Equation 2.17), due to its ubiquity in previous PbRL literature and its desirable properties discussed in Section 2.5. Since, differently from the classical Bradley-Terry model application domain, our problem setting requires the expert model γ to be 4-categorical, the most straightforward choice is the multinomial logit model [57] with four classes:

$$\gamma_y(\Delta \mathbf{U}) = \frac{\exp f_y}{\sum_{y' \in \mathcal{Y}} \exp f_{y'}}, y \in \mathcal{Y}, \quad (5.1)$$

where $f_y, y \in \mathcal{Y}$ denote the logits of the four classes. These logits depend on the multi-dimensional utility difference $\Delta \mathbf{U}$, i.e., $f_y = f_y(\Delta \mathbf{U}), y \in \mathcal{Y}$ and such functions need to be carefully chosen to make sure γ is compliant with Definition 4.1.

5.1.1. Derivation of Model A

We start by designing models that present all the properties of the rational multi-objective expert model, setting aside, for now, the limitations in the optimization procedure highlighted by Theorem 4.1. In this Section, we present the first model choice, i.e., Model A, starting with an intuitive description of the mathematical objects involved in the definition of this model, and then proceeding to define them in a formal way.

Intuitive Explanation

We decide to reason, for simplicity, in the case $d = 2$ (therefore the utility is a 2-dimensional vector $\mathbf{U}(\tau) = (U_1(\tau), U_2(\tau))^\top$), and consider a single trajectory pair $\{\tau_1, \tau_2\}$. It is easier to reason in the $\Delta U_1/\Delta U_2$ plane, as we notice that a point $\Delta \mathbf{U}$ in this plane is exactly the input to an expert model $\gamma(\Delta \mathbf{U})$.

Conditions 1, 2 and 5 of Definition 4.1 prescribe particular values of the γ components along the line $\mathbf{1}_d \cdot t$, $t \in \mathbb{R}$, that, in case $d = 2$, is the bisector $\Delta U_1 = \Delta U_2$. In particular, the projection of the point on the bisector is associated with a measure of strict preference for the pair (classes \succ and \prec), since in $(+\infty, +\infty)^\top$ the norm of such projection is infinite and the sign is positive, giving probability $\simeq 1$ to the \succ class. The same, with negative sign, happens at $(-\infty, -\infty)^\top$, shifting the probability mass to the \prec class.

For the incomparability class, Condition 5 prescribes the incomparability score to be minimum along the bisector and Condition 4 places its maximum at $(+\infty, -\infty)^\top$ and $(-\infty, +\infty)^\top$, therefore a simple measure of incomparability for a point $(\Delta U_1, \Delta U_2)^\top$ would be its distance from the bisector.

The remaining score, i.e., the indifference one, is expected to take its maximum value in the origin $(0, 0)^\top$. Since all the scores are arguments of a softmax, one can simply put a constant indifference score β_{\asymp} such that, when all the other scores are low (and this happens only around the origin, where $\beta_{\succ}((0, 0)^\top) = \beta_{\prec}((0, 0)^\top) = \beta_{\asymp}((0, 0)^\top) = 0$) the probability mass is shifted towards \asymp .

The following derivation follows the same steps for deriving the score functions $f_{\succ}(\Delta \mathbf{U})$, $f_{\prec}(\Delta \mathbf{U})$, $f_{\asymp}(\Delta \mathbf{U})$ and $f_{\parallel}(\Delta \mathbf{U})$ in the case of a generic dimensionality d . The bisector is now the standard diagonal in \mathbb{R}^d and we show that the distance of the point $\Delta \mathbf{U}$ from this line is proportional to the standard deviation of $\Delta \mathbf{U}$.

Formal Derivation

For notational convenience, we will, in the following, denote the multi-dimensional utility difference vector as $\mathbf{p} \in \mathbb{R}^d$. Let $\mathbf{v} \in \mathbb{R}^d$ be the vector defined as:

$$\mathbf{v} = \left(\frac{1}{\sqrt{d}} \quad \frac{1}{\sqrt{d}} \quad \dots \quad \frac{1}{\sqrt{d}} \right)^\top = \mathbf{1}_d \cdot \frac{1}{\sqrt{d}} \in \mathbb{R}^d. \quad (5.2)$$

Then, the standard diagonal in \mathbb{R}^d can be defined as the span of \mathbf{v} : $\mathcal{D} = \{\mathbf{r} \in \mathbb{R}^d \mid \mathbf{r} = \mathbf{v} \cdot t, t \in \mathbb{R}\}$.

We can write the projection of the point \mathbf{p} onto the standard diagonal in \mathbb{R}^d as the point $\mathbf{g} = \bar{t} \cdot \mathbf{v}$ such that \mathbf{v} is orthogonal to $(\mathbf{g} - \mathbf{p})$:

$$\mathbf{v}^\top (\mathbf{g} - \mathbf{p}) = \mathbf{v}^\top (\mathbf{v} \cdot \bar{t} - \mathbf{p}) = 0, \quad (5.3)$$

from which, solving for \bar{t} :

$$\mathbf{v}^\top (\mathbf{v} \bar{t} - \mathbf{p}) = \bar{t} - \mathbf{v}^\top \mathbf{p} = 0 \implies \bar{t} = \mathbf{1}_d^\top \cdot \frac{1}{\sqrt{d}} \cdot \mathbf{p}, \quad (5.4)$$

obtaining:

$$\mathbf{g} = \bar{t} \cdot \mathbf{v} = \left(\mathbf{1}_d^\top \cdot \frac{1}{\sqrt{d}} \cdot \mathbf{p} \right) \cdot \mathbf{v} = \left(\mathbf{1}_d^\top \cdot \frac{1}{\sqrt{d}} \right) \cdot \mathbf{p} \cdot \left(\mathbf{1}_d \cdot \frac{1}{\sqrt{d}} \right) = \frac{1}{d} \cdot \mathbf{1}_d \mathbf{1}_d^\top \cdot \mathbf{p}. \quad (5.5)$$

We note that we can further manipulate \mathbf{g} to obtain a vector having all components equal to the mean of the \mathbf{p} vector (denoted as $\bar{\mathbf{p}}$):

$$\mathbf{g} = \frac{1}{d} \cdot \mathbf{1}_d \mathbf{1}_d^\top \cdot \mathbf{p} = \frac{1}{d} \cdot \mathbf{1}_d \cdot (\mathbf{1}_d^\top \cdot \mathbf{p}) = \frac{1}{d} \mathbf{1}_d \sum_{i=1}^d p_i = \left(\frac{1}{d} \sum_{i=1}^d p_i \right) \mathbf{1}_d = \bar{\mathbf{p}}. \quad (5.6)$$

We can now write the distance from the bisector as:

$$f_{\parallel}(\mathbf{p}) := \|\mathbf{g} - \mathbf{p}\|_2 = \|\bar{\mathbf{p}} - \mathbf{p}\|_2 = \sqrt{\sum_{i=1}^d (p_i - \bar{p})^2} = \sqrt{d} \cdot std(\mathbf{p}). \quad (5.7)$$

where $std(\mathbf{x}) = \frac{\|\mathbf{x} - \bar{\mathbf{x}}\|}{\sqrt{d}}$ is the standard deviation of the vector \mathbf{x} .

If we now write the distance of \mathbf{g} (the projection) from the origin we obtain:

$$N(\mathbf{p}) := \|\mathbf{g}\|_2 = \left\| \left(\frac{1}{d} \sum_{i=1}^d p_i \right) \cdot \mathbf{1}_d \right\| = \frac{1}{d} \left| \sum_{i=1}^d p_i \right| \cdot \|\mathbf{1}_d\|_2 = \frac{1}{\sqrt{d}} \left| \sum_{i=1}^d p_i \right|. \quad (5.8)$$

We can define the signed norm of the point \mathbf{g} , very similarly to $N(\mathbf{p})$:

$$f_{\succ}(\mathbf{p}) := \frac{1}{\sqrt{d}} \sum_{i=1}^d p_i = -f_{\prec}(\mathbf{p}). \quad (5.9)$$

The last quantity of interest is, as already mentioned, a (learnable) constant, that will act as a measure of indifference for point \mathbf{p} :

$$f_{\asymp}(\mathbf{p}) := K. \quad (5.10)$$

A graphical example in the 2-dimensional case is presented in Figure 5.1.

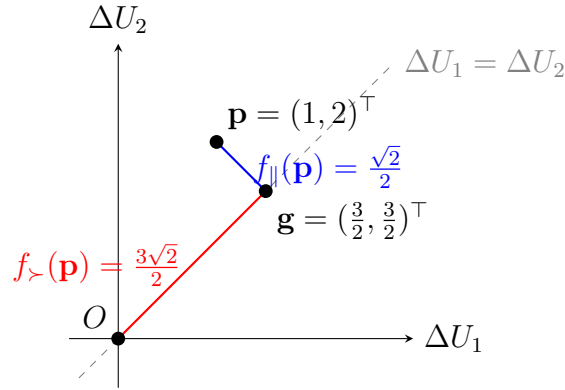


Figure 5.1: Representation of the values f_{\parallel} and f_{\succ} on the $(\Delta U_1, \Delta U_2)$ plane.

Final Model

Having defined f_{\succ} , f_{\prec} , f_{\asymp} and f_{\parallel} , the idea is to use these functions applied to $\Delta \mathbf{U}$ as logits for the corresponding 4 classes of our problem. The resulting model is obtained by simply putting these logits in a 4-class multinomial logit model (or, equivalently, by applying a *softmax* function to the logits):

$$\gamma_y(\Delta \mathbf{U}) = \frac{\exp f_y(\Delta \mathbf{U})}{\sum_{y' \in \mathcal{Y}} \exp f_{y'}(\Delta \mathbf{U})}, y \in \mathcal{Y}. \quad (5.11)$$

Additional Details

Implementation details We notice that $f_{\parallel}(\Delta\mathbf{U} = t \cdot \mathbf{1}_d) = 0, \forall t \in \mathbb{R}$, this is contrasting with property 5 of the rational multi-objective expert model definition since we have $\gamma_{\parallel}(t \cdot \mathbf{1}_d) > 0, \forall t \in \mathbb{R}$. However, this problem would be solved only for values of f_{\parallel} on the standard diagonal equal to $-\infty$. We cannot completely solve this problem without fundamentally altering the model, but we can subtract a fixed constant $\alpha > 0$ to f_{\parallel} in such a way that, on the standard diagonal, the value of $P(\tau_1 \parallel \tau_2) = \gamma_{\parallel}(\Delta\mathbf{U})$ is very close to 0. For $\alpha \rightarrow +\infty$, we have $f_{\parallel}(\Delta\mathbf{U} = t \cdot \mathbf{1}_d) \rightarrow -\infty$ and γ becomes compliant to the rational multi-objective expert model.

In real-life applications, the expert could lean towards a preference even if the multi-objective real reward difference $\Delta\hat{\mathbf{U}}(\tau_1, \tau_2)$ presents very large, positive values on some objectives and small, negative values on others. In other terms, this is the case in which τ_1 is much better than τ_2 on some objectives, but slightly worse on others. This behavior is highly subjective and cannot be easily formalized. We can visualize these areas in the $d = 2$ case in Figure 5.2. Fortunately, subtracting $\alpha > 0$ to f_{\parallel} has the side effect of shifting the probability mass towards strict preferences in these regions of the $\Delta\mathbf{U}$ -space, solving both problems.

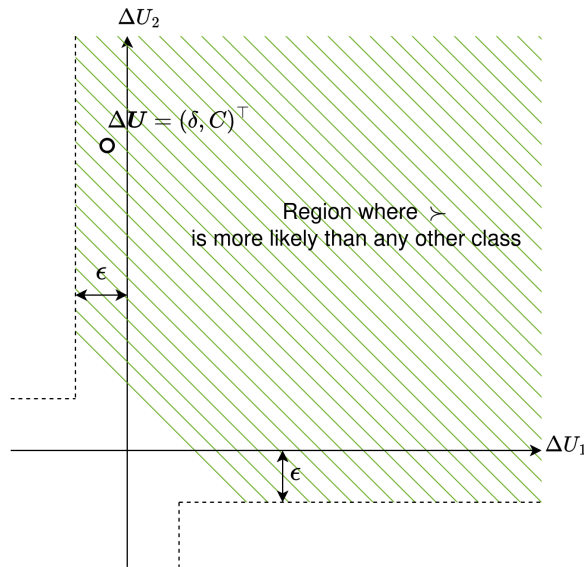


Figure 5.2: Visualization of the area in which it is desirable that $P(\tau_1 \succ \tau_2) > P(\tau_1 \succcurlyeq \tau_2), \forall y \in \{\prec, \asymp, \parallel\}$. The point $\Delta\mathbf{U} = (\delta, C)^\top$ should be given a high probability of strict preference because $\delta \simeq 0$ and $C \gg 0$.

From now on, we will consider this slight variation of Model A, in which the logit function

for the incomparability class is replaced by:

$$f_{\parallel}^{\alpha}(\Delta U) := \sqrt{d} \cdot \text{std}(\Delta U) - \alpha, \quad (5.12)$$

where α is a parameter to be estimated.

Remark 5.1 (The model reduces to the BT model for $d=1$). *We note that, for scalar utilities, we are in the case $\Delta U \in \mathbb{R}$, and the logit functions become:*

$$\begin{aligned} f_{\succ}(\Delta U) &= \Delta U, \\ f_{\prec}(\Delta U) &= -\Delta U, \\ f_{\asymp}(\Delta U) &= K, \\ f_{\parallel}^{\alpha}(\Delta U) &= \text{std}(\Delta U) - \alpha = \|\Delta U - \Delta U\| - \alpha = -\alpha. \end{aligned} \quad (5.13)$$

Where K is a constant that is directly proportional to the percentage of indifferences in ζ , and α is a constant large enough to give a value close to 0 to the \parallel class on the standard diagonal.

Therefore, the model reduces, in the limit $\alpha \rightarrow +\infty$, to the BT model with an additional indifference class, very similarly to the model described in Equation 3.2:

$$\begin{aligned} P(\tau_i \succ \tau_j) &= \gamma_{\succ}(\Delta U) \approx \frac{e^{\Delta U}}{e^{\Delta U} + e^{-\Delta U} + e^K}, \\ P(\tau_i \asymp \tau_j) &= \gamma_{\asymp}(\Delta U) \approx \frac{e^K}{e^{\Delta U} + e^{-\Delta U} + e^K}, \\ P(\tau_i \parallel \tau_j) &= \gamma_{\parallel}(\Delta U) \approx 0. \end{aligned} \quad (5.14)$$

In the case of $K \rightarrow -\infty$, i.e., there are no indifference samples in the dataset, the model is equivalent to the BT one, except for a scaling factor in the learned utilities:

$$P(\tau_i \succ \tau_j) = \gamma_1(\Delta U) = \frac{e^{\Delta U}}{e^{\Delta U} + e^{-\Delta U}} = \frac{1}{1 + e^{-2\Delta U}}. \quad (5.15)$$

Optimizing the model In the case of a linear reward model, i.e., $\mathbf{U}(\tau) = \mathbf{W}\phi(\tau)$ the optimizable parameters are the matrix $\mathbf{W} \in \mathbb{R}^{d \times k}$, the bias term α for the incomparability class and the indifference class constant $K \in \mathbb{R}$. Therefore, $d \cdot k + 2$ parameters. One possible choice - and the one adopted in our experiments - is to find the optimal values for the parameters using MLE, or, equivalently, minimizing the negative log-likelihood

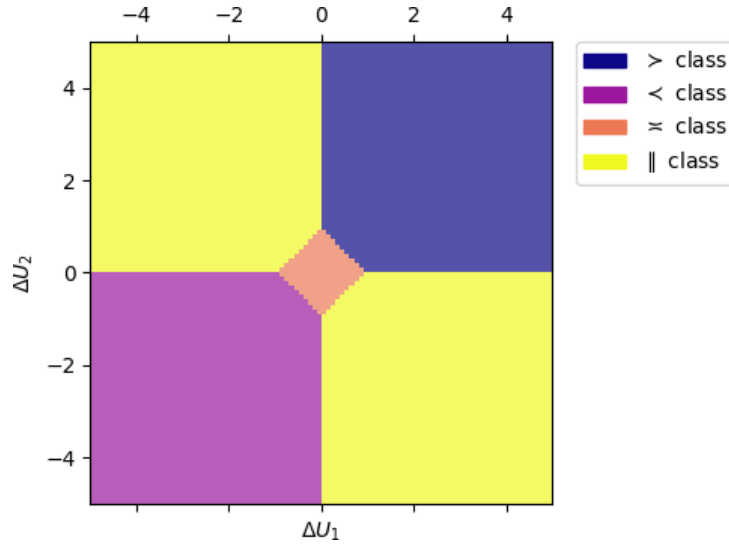


Figure 5.3: $\arg \max_y \gamma_y((\Delta U_1, \Delta U_2)^\top)$ in the case $K = 1$, $\alpha = 0$ (Model A).

function:

$$\begin{aligned} \hat{\mathbf{W}}^{MLE}, \hat{K}^{MLE}, \hat{\alpha}^{MLE} &= \arg \min_{\mathbf{W}, K, \alpha} l(\mathbf{W}, K, \alpha; \zeta) \\ &= \arg \min_{\mathbf{W}, K, \alpha} \left\{ - \sum_{i=1}^N \log \gamma_{y_i}(\mathbf{W}(\phi(\tau_{i1}) - \phi(\tau_{i2}))) \right\}. \end{aligned} \quad (5.16)$$

As already noted in Remark 4.1, this loss function is not convex in \mathbf{W} and K for all possible datasets ζ . This can be graphically seen by plotting, for example, the graph of $z(W_{1,1}, W_{1,2})$:

$$\begin{aligned} z(W_{1,1}, W_{1,2}) &:= l \left(\begin{pmatrix} W_{1,1} & W_{1,2} \\ 0 & 0 \end{pmatrix}, 1, 1; \zeta \right), \\ \zeta &= \{(\tau_1, \tau_2, ||)\}, \quad \phi(\tau_1) = (1, 1)^\top, \quad \phi(\tau_2) = (0, 0)^\top. \end{aligned} \quad (5.17)$$

The graph of $z(W_{1,1}, W_{1,2})$ is shown in Figure 5.4. As it can be seen, the restriction of l to these two variables is clearly non convex, therefore $l(\mathbf{W}, K, \alpha; \zeta)$ is non convex.

Standard methods of *non-convex optimization* [28] can be employed; however, few theoretical guarantees can be provided on the computational complexity and on the goodness of the found parameter estimates.

5.1.2. Derivation of Model B

In this section, we present another possibility for an expert model complying with all properties of the rational multi-objective expert model, showing that we get a semantically

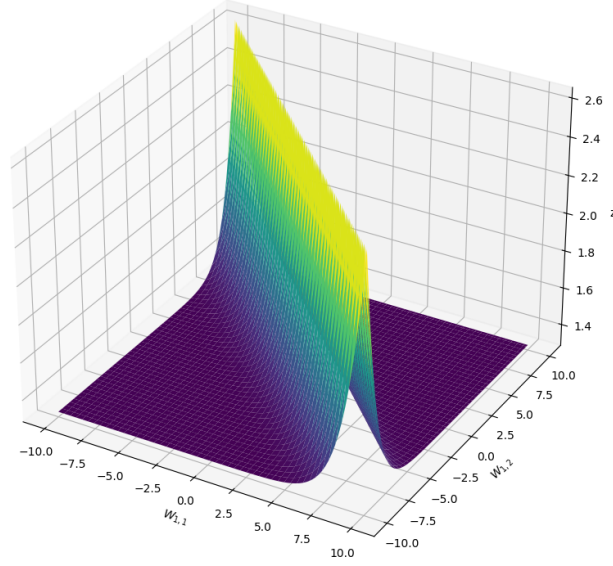


Figure 5.4: Non convexity of negative log-likelihood of Model A.

equivalent model to Model A.

Intuitive Explanation

We now give an intuitive explanation for the mathematical objects involved in the definition of this model's logit functions. We reason in the case of $\Delta\mathbf{U}$ limited to values in $[-1, 1]$ along all of the d dimensions (but the same reasoning applies to the case of unbounded values for the utility difference). In other words, $\Delta\mathbf{U} \in [0, 1]^d$, i.e., $\Delta\mathbf{U}$ takes values in the d dimensional hypercube centered in $\mathbf{0}_d$ and having edge length equal to 2. We note that we can logically associate the various orthants with a class in \mathcal{Y} : the score associated with preference needs to be maximal in the orthants corresponding to all components of $\Delta\mathbf{U}$ being positive or negative, i.e., $\mathcal{D}_{\succ} = \{\Delta U_i \geq 0, \forall i \in \llbracket d \rrbracket\}$ and $\mathcal{D}_{\prec} = \{\Delta U_i \leq 0, \forall i \in \llbracket d \rrbracket\}$. In all other orthants, i.e., the ones with some positive and some negative components, the maximal score should be the one corresponding to the incomparability class, since it means we are in a situation with contrasting objectives for the two trajectories. The indifference is maximal around the origin, while the scores for the other classes are higher the more the point is closer to the vertices of the hypercube.

If we now consider the line segments joining the origin with all the vertices of the hypercube, and project the point $\Delta\mathbf{U}$ on those segments, we get an estimate of the class scores for that point (if we sum all the scores for the incomparability class).

Formal Definition

We can define the following functions:

$$\begin{aligned}
 f_{\succ}(\Delta \mathbf{U}) &:= \mathbf{1}_d^\top \cdot \Delta \mathbf{U} = \sum_{i=1}^d \Delta U_i, \\
 f_{\prec}(\Delta \mathbf{U}) &:= -\mathbf{1}_d^\top \cdot \Delta \mathbf{U} = -\sum_{i=1}^d \Delta U_i, \\
 f_{\asymp}(\Delta \mathbf{U}) &:= K, \\
 f_{\parallel}^{(j)}(\Delta \mathbf{U}) &:= \mathbf{v}^{(j)\top} \cdot \Delta \mathbf{U} = \sum_{i=1}^d v_i^{(j)} \Delta U_i,
 \end{aligned} \tag{5.18}$$

where $\{\mathbf{v}^{(j)}\}_{j \in \llbracket 2^d - 2 \rrbracket} = \{-1, 1\}^d \setminus \{\mathbf{1}_d, -\mathbf{1}_d\}$, i.e., the set of all vertices of the d -dimensional hypercube centered in $\mathbf{0}_d$ and having edges of length 2, except from the vertices $\mathbf{1}_d$ and $-\mathbf{1}_d$. The index numbering of such vectors $j \in \llbracket 2^d - 2 \rrbracket$ is arbitrary.

Final Model

We can now define the expert model γ by again utilizing the *softmax* function applied to the logits of the classes, but this time considering a class for every function and summing the logits of the incomparability class to obtain $\gamma_{\parallel}(\cdot)$. For notational convenience, let the denominator of the softmax be:

$$DEN(\Delta \mathbf{U}) := \sum_{y \in \{\succ, \prec, \asymp\}} \exp f_y(\Delta \mathbf{U}) + \sum_{j=1}^{2^d - 2} \exp f_{\parallel}^{(j)}(\Delta \mathbf{U}). \tag{5.19}$$

Then, we can define the expert model γ as:

$$\begin{aligned}
 \gamma_y(\Delta \mathbf{U}) &= \frac{\exp f_y(\Delta \mathbf{U})}{DEN(\Delta \mathbf{U})}, y \in \{\succ, \prec, \asymp\}, \\
 \gamma_{\parallel}(\Delta \mathbf{U}) &= \frac{\sum_{j=1}^{2^d - 2} \exp f_{\parallel}^{(j)}(\Delta \mathbf{U})}{DEN(\Delta \mathbf{U})}.
 \end{aligned} \tag{5.20}$$

Additional Details

Implementation Details Similarly to Model A, one issue with this model is that it doesn't account for a "margin" in allowing small incomparabilities when one of the two trajectories is clearly superior along some dimensions of the utility. To solve this issue, we can add a (learnable) constant α to f_{\succ} and f_{\prec} . We will refer to this slight modification

of the model from now on.

Remark 5.2 (The model reduces to the BT model for $d=1$). *We note that, for scalar utilities, we, again, obtain a generalization of the Bradley-Terry model with ties formulated by [48]. Since $d = 1$, we actually have no incomparability functions to consider ($\{f_{\parallel}^{(j)}\}_{j=1}^{2^1-2} = \emptyset$), and the other functions become:*

$$\begin{aligned} f_{\succ}(\Delta U) &= \Delta U + \alpha, \\ f_{\prec}(\Delta U) &= -\Delta U + \alpha, \\ f_{\asymp}(\Delta U) &= K, \end{aligned} \tag{5.21}$$

equivalently to Equation 5.13. Therefore, this model is equivalent to Equation 5.14 and, if $K \rightarrow -\infty$, it becomes equivalent to the BT model, except for translating and rescaling the utilities (see Equation 5.15).

Optimizing the model Similarly to Model A, if we employ a linear utility function $U(\tau) = \mathbf{W}\phi(\tau)$, the number of parameters to estimate is $d \cdot k + 2$. Once again, we can minimize the negative log-likelihood function (see Equation 5.16) to find parameter estimates $\hat{\mathbf{W}}^{MLE}$, \hat{K}^{MLE} and $\hat{\alpha}^{MLE}$. Even for this model, the log likelihood is not convex for every possible dataset ζ .

Equivalence with Model A and computational complexity This model achieves the same degree of expressiveness (defined as the degree of compliance with the rational multi-objective expert model properties) as Model A. Although its construction may result more intuitive to the reader than that of Model A, we note that computing the model output for a comparison (i.e., computing $\gamma(\Delta \mathbf{U})$) requires $O(2^d)$ operations due to the fact that we need to compute the logit function for each of the $2^d - 2$ incomparability classes. Therefore, from a computational complexity point of view, Model A is to be preferred as this operation runs in $O(d)$.

5.1.3. Derivation of Model C

In this section, we address a question that may naturally arise after the presentation of Theorem 4.1 (and in particular of Remark 4.1) and the derivations of Models A and B: how many (and which) properties of the rational multi-objective expert model (Definition 4.1) must be violated in order to obtain a model that is optimizable via MLE with a convex loss?

As can be seen in the proof of Theorem 4.1, the most problematic properties are the ones regarding the incomparability class, namely property 4 and 5, which we restate:

4. $\forall \mathbf{v} \in \{-1, 1\}^d$ s.t. $\mathbf{v} \neq \pm \mathbf{1}_d$: $\gamma(\Delta \mathbf{U}) \rightarrow (0, 0, 0, 1)^\top$ as $\Delta \mathbf{U} \rightarrow \mathbf{v} \odot (+\infty, \dots, +\infty)^\top$,
5. $\forall t \in \mathbb{R}$: $\gamma_4(t \cdot \mathbf{1}_d) = 0$.

In this section, we illustrate our attempt in deriving a model with convex negative log-likelihood and such that all properties besides 4 and 5 are respected. We conclude by outlining its limitations in terms of expressivity.

Theorem 5.1 (Convexity of negative log-likelihood for multinomial logit model). *Let $l: \mathbb{R}^d \rightarrow \mathbb{R}$, $d \geq 1$ be the function defined as:*

$$l(\mathbf{x}) := -\log \left(\frac{\exp(\mathbf{a}_1^\top \mathbf{x} + b_1)}{\sum_{i=1}^n \exp(\mathbf{a}_i^\top \mathbf{x} + b_i)} \right), \quad (5.22)$$

then, l is convex in its argument \mathbf{x} for all choices of $\mathbf{a}_i \in \mathbb{R}^d, b_i \in \mathbb{R}, n \in \mathbb{N} \setminus \{0\}$.

Proof. We can equivalently express $l(\mathbf{x})$ as:

$$l(\mathbf{x}) = -\mathbf{a}_1^\top \mathbf{x} - b_1 + \log \left(\sum_{i=1}^n \exp(\mathbf{a}_i^\top \mathbf{x} + b_i) \right). \quad (5.23)$$

We note that $-\mathbf{a}_1^\top \mathbf{x} - b_1$ is convex in \mathbf{x} since it is a linear affine function of \mathbf{x} (therefore, both convex and concave). $\log(\sum_{i=1}^n \exp(z_i))$ is convex and non-decreasing in each argument z_i , therefore $\log(\sum_{i=1}^n \exp(f_i(\mathbf{x})))$ is convex whenever $f_i(\mathbf{x})$ are convex in \mathbf{x} , $\forall i \in \llbracket n \rrbracket$ [9, Section 3.2.4]. Since the sum of two convex functions is convex [9, Section 3.2.1], l is convex in its argument \mathbf{x} . \square

Given the result of Theorem 5.1, and noticing that l is the negative log-likelihood function of a multinomial logit model in which each logit is obtained by an affine transformation of the input vector \mathbf{x} , we can think of a model in which the logit functions $f_{\succ}, f_{\prec}, f_{\asymp}$ and f_{\parallel} are affine. Convexity in \mathbf{W} is then guaranteed since $\Delta \mathbf{U} = \mathbf{W}(\phi(\tau_1) - \phi(\tau_2))$ is a linear transformation and as such it preserves convexity [9, Section 3.2.2].

As for classes \succ and \prec , their corresponding logit functions as defined in Model A and Model B are already linear functions of $\Delta \mathbf{U}$, therefore, not problematic. The use of the standard deviation (as in Model A) as the logit function for the incomparability class is problematic, since this function is not linear on \mathbb{R}^d and it cannot be linearized without error. We can visualize this by plotting the graph of $std(\mathbf{x})$ in the case $d = 2$ (see Figure

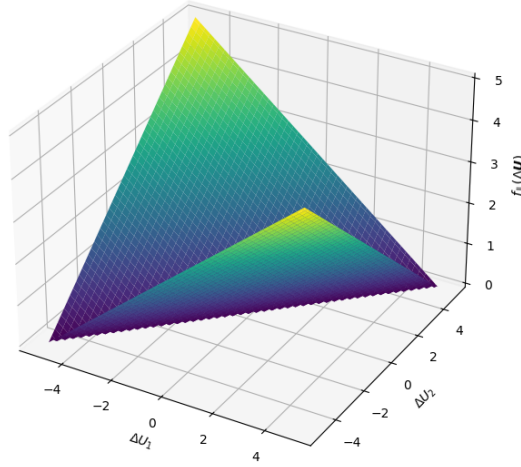


Figure 5.5: Plot of $f_{\parallel}(\Delta \mathbf{U}) = \text{std}(\Delta \mathbf{U})$ for $d = 2$

5.5)

Given the function $f(\mathbf{x}) = \text{std}(\mathbf{x})$, every possible linear affine approximator $g(\mathbf{x}) = \mathbf{a}^\top \mathbf{x} + b$ is such that the absolute error $J(g, \mathbf{x}) = |f(\mathbf{x}) - g(\mathbf{x})|$ is unbounded for some choice of $\mathbf{x} \in \mathbb{R}^d$. We can think of the function g that minimizes the least-squares error in an hypercube centered in $\mathbf{0}_d$ and having edge length of $2T$ (i.e., $\mathcal{D}_T^d := [-T, T]^d$):

$$\hat{\mathbf{a}}, \hat{b} = \arg \min_{\mathbf{a}, b} \int_{\mathcal{D}_T^d} (\sqrt{d} \cdot \text{std}(\mathbf{x}) - (\mathbf{a}^\top \mathbf{x} + b))^2 d\mathbf{x}, \quad (5.24)$$

obtaining $\hat{\mathbf{a}} = \mathbf{0}_d, \hat{b} = \frac{T}{2}$. We decide, since the utility differences are not bounded in the general case, to utilize a learnable constant as the indifference class logit function $f_{\parallel}(\Delta \mathbf{U}) = \alpha$.

This choice is problematic because the indifference (\asymp) class' logit function was defined as a constant in Model A and Model B. No other linear f_{\asymp} is such that Condition 3 of Definition 4.1 is achieved having defined the aforementioned f_{\succ}, f_{\prec} and f_{\parallel} . In fact, we conjecture that one cannot simultaneously achieve Conditions 3 and Conditions 4, 5 of the rational multi-objective expert model using linear logit functions $f_{\succ}, f_{\prec}, f_{\asymp}$ and f_{\parallel} .

Final Model

One possible solution is to discard the possibility of having indifferences in the dataset and derive an expert model $\gamma : \mathbb{R}^d \rightarrow \Delta^2$, therefore a 3-categorical probability function, such that $\gamma(\Delta \mathbf{U}) = (\gamma_{\succ}(\Delta \mathbf{U}), \gamma_{\prec}(\Delta \mathbf{U}), \gamma_{\parallel}(\Delta \mathbf{U}))^\top$ by, again, applying a *softmax* function

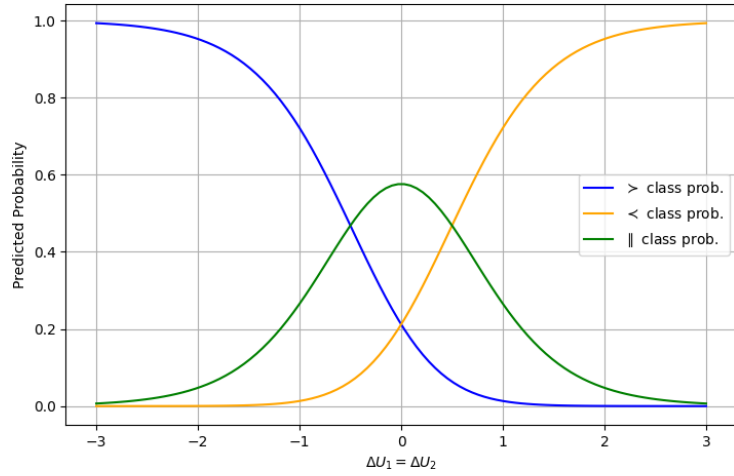


Figure 5.6: Output of Model C along the standard diagonal ($\alpha = 1$).

to the $f_{>}(\Delta\mathbf{U})$, $f_{<}(\Delta\mathbf{U})$ and $f_{||}(\Delta\mathbf{U})$ defined before:

$$\gamma_y(\Delta\mathbf{U}) = \frac{\exp f_y(\Delta\mathbf{U})}{\sum_{y' \in \{>, <, ||\}} \exp f_{y'}(\Delta\mathbf{U})}. \quad (5.25)$$

Other possibilities are:

1. Considering samples from the indifference (\asymp) class instead of the incomparability ($||$) one, and swapping $\gamma_{||}$ with γ_{\asymp} (since the two classes' prediction regions overlap in this model).
2. Consider a class comprising both indifferences and incomparabilities.

Considerations

As proved in Theorem 4.1, and as remarked by the derivation of this model, designing an expert model γ such that: (a) the log-likelihood of γ on any dataset ζ constructed as described in Section 4.2.1 is convex in its parameters and (b) it complies with the definition of rational multi-objective expert model, is proved to be impossible.

In fact, this model is equivalent to the Bradley-Terry model with Ties (Equation 3.2), in which we use as scalar utility $U(\tau)$ the simple scalarization $\mathbf{1}_d^\top \cdot \mathbf{U}(\tau)$ and in which the incomparability class is swapped with the indifference class. For visualization purposes, see Figure 5.6 for the output of Model C on the reduction to the $d = 1$ case and Figure 5.7 for the *argmax* of the predicted classes over the utility space in the case $d = 2$.

By employing this model, we are implicitly scalarizing the utility vector with weights $\mathbf{1}_d$, therefore making an assumption on the relative importance of the various objectives of

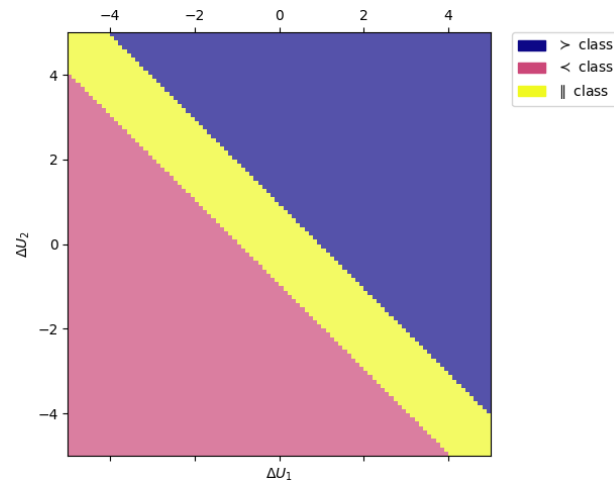


Figure 5.7: $\arg \max_y \gamma_y((\Delta U_1, \Delta U_2)^\top)$ in the case $\alpha = 1$ (Model C).

the problem. Even though this is commonly done in previous works, this is a blind choice that presents many issues, as thoroughly discussed in [25].

6 | Experiments

In this chapter, we validate the models proposed in Chapter 5 through various experiments. Firstly, in Section 6.1 we describe technical choices that have been done in implementing and running the code. Then, in Section 6.2, we present the results of optimizing the models' parameters on a synthetic dataset. Finally, in Section 6.3 we present and comment various experiments conducted on a gridworld with multi-dimensional true reward.

6.1. Implementation Details

In this Section, we discuss the choices and details required for implementing and evaluating the models introduced in Chapter 5.

Linear model As already briefly discussed, we develop our code employing a reward model that is linear in the feature mapping vectors, i.e., $\mathbf{U}(\tau) = \mathbf{W} \cdot \phi(\tau)$. This is a simplifying choice made for the sake of easier interpretability of the results, and non-linear extensions of the reward model are straightforward.

BFGS/L-BFGS optimization algorithm One particular aspect that has been deliberately omitted until now is the specific algorithm for the minimization of the negative log-likelihood function over a specific dataset, in order to obtain the maximum likelihood estimation for the model's parameters. We adopt the Broyden–Fletcher–Goldfarb–Shanno [BFGS, 39, Section 6.1] algorithm for unconstrained minimization, as this method is one of the state of the art algorithms for settings in which there are no guarantees on the convexity of the objective function, while providing a lower computation time for solving the optimization problem with respect to alternative choices. In the cases where the number of parameters to be estimated is substantial, we employ its limited-memory version, Limited Memory-BFGS [L-BFGS, 13].

Mini-batch gradient descent Additional implementations of Model A and Model B were developed in PyTorch [44], which provides efficient tensor computations. Training

was performed using mini-batch gradient descent with the Adam [31] optimizer.

The optimization parameters were set to the following values:

- $learning_rate = 10^{-4}$
- $batch_size = 128$
- $epochs = 150$
- $\beta_1 = 0.9, \beta_2 = 0.999, \epsilon = 10^{-8}$ (Adam parameters)

Choice of d In real world applications, the true number of underlying objectives the expert uses to label pairs of trajectories (\hat{d}) is not known in advance to the algorithm developer. In fact, in some real-world applications, it may happen that the human experts themselves are not aware of the number of objectives in advance. It is clear that this issue is out of the scope of these experiments, as the main focus is now the model’s parameters learning process; therefore, we artificially set the number of dimensions of \mathbf{U} to \hat{d} , i.e., $d = \hat{d}$. We briefly discuss the problem of choosing the value of d in real applications in Section 7.1.

Evaluation Metrics Since this work focuses solely on the reward modeling step, we decide to resort to evaluating the learned reward models in isolation, without learning a policy or a set of policies afterward and evaluating those. We evaluate the reward model by (a) using the learned \mathbf{W}^{MLE} to predict a class for each of the dataset trajectory pair, and then compare them with the true labels (therefore treating the problem as a classification one, evaluating precision, recall and F1-score) and (b) comparing the probability outputs directly (i.e., $\gamma(\hat{\mathbf{W}}\Delta\phi)$ and $\gamma(\mathbf{W}^{MLE}\Delta\phi)$), by evaluating the average Jensen-Shannon divergence (see Equation 2.4) over all samples in the dataset.

Hardware and Software specifics The code used for the following experiments has been run on an Intel(R) Xeon(R) Gold 6238R CPU @ 2.20GHz processor, with 256GB of RAM. The algorithms are implemented in *Python 3.13.9* and run on a machine with operative system *Ubuntu 20.04.6 LTS*.

6.2. Preliminary Validation on Synthetic Datasets

Description

In this experimental setting, we consider an expert model that employs the same model class that is used to learn the reward model parameter matrix \mathbf{W} . A real linear reward model is employed by the expert (i.e., $\hat{U}(\tau) = \hat{\mathbf{W}} \cdot \phi(\tau)$). Of course, the expert also knows the other real parameters of the models: $\hat{\alpha}$ for Model C, \hat{K} and $\hat{\alpha}$ for Model A and Model B. Since the focus of these experiments is to verify the correctness of the reconstructed matrix \mathbf{W}^{MLE} , we assume that the learner has knowledge of both α and K , thus setting them to their real values $\hat{\alpha}$ and \hat{K} , and keeping them fixed to such values.

We construct the preference dataset as follows. First, we sample $N_P = 10000$ points $\{\phi_i\}_{i \in [N]}$ in the trajectory feature mapping space, i.e., \mathbb{R}^k , from the uniform distribution in $[0, 1]^k$. Then, for $N = 5000$ times, we randomly select two feature vectors (ϕ_i, ϕ_j) , $i, j \in [N]$ and the expert evaluates the pair, obtaining $y_{ij} \sim \hat{\gamma}(\hat{\mathbf{W}}(\phi_i - \phi_j))$. Finally, we add such pairs and labels to the dataset. We employ $N_{TR} = 4000$ samples for training the models and $N_{TE} = 1000$ samples for testing and displaying results.

For each model, we evaluate the reconstructed linear mapping $\mathbf{W}^{MLE} \in \mathbb{R}^{d \times k}$ and plot the test samples' reconstructed utility difference $\Delta U_{ij}^{MLE} := \mathbf{W}^{MLE}(\phi(\tau_i) - \phi(\tau_j)) \in \mathbb{R}^d$, comparing them and their (sampled) labels to the real ones. Moreover, for each combination of model employed by the expert and model used to learn \mathbf{W} , we analyze the average *Jensen-Shannon divergence* metric for the distribution of the labels.

Results

We display experimental results in the case $d = 2$, $k = 3$. Let the real parameter matrix be:

$$\hat{\mathbf{W}} = \begin{pmatrix} 1 & 2 & 3 \\ 4 & -1 & 5 \end{pmatrix} \quad (6.1)$$

We now report the matrices reconstructed by the three models and discuss upon them.

Model A The learned parameter matrix is:

$$\mathbf{W}_A^{MLE} \simeq \begin{pmatrix} 1.353 & 2.210 & 3.335 \\ 5.332 & -1.002 & -6.163 \end{pmatrix} \quad (6.2)$$

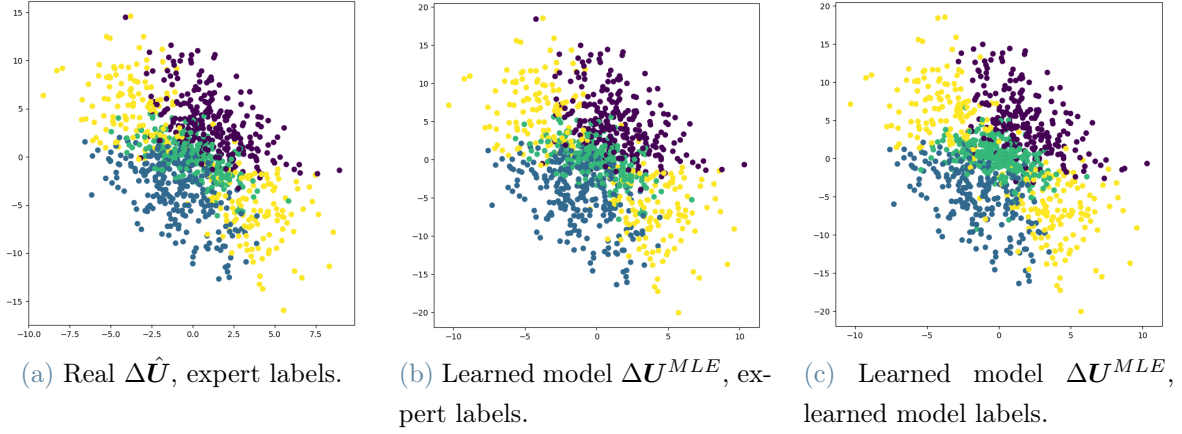


Figure 6.1: Test set pairs real and learned mappings for Model A in the $\Delta U_1/\Delta U_2$ space. Labels: preference (\succ), inverted preference (\prec), indifference (\asymp), incomparability (\parallel).

Model B The learned parameter matrix is:

$$\mathbf{W}_B^{MLE} \simeq \begin{pmatrix} 3.313 & 4.264 & 5.881 \\ 10.798 & -1.509 & -11.502 \end{pmatrix} \quad (6.3)$$

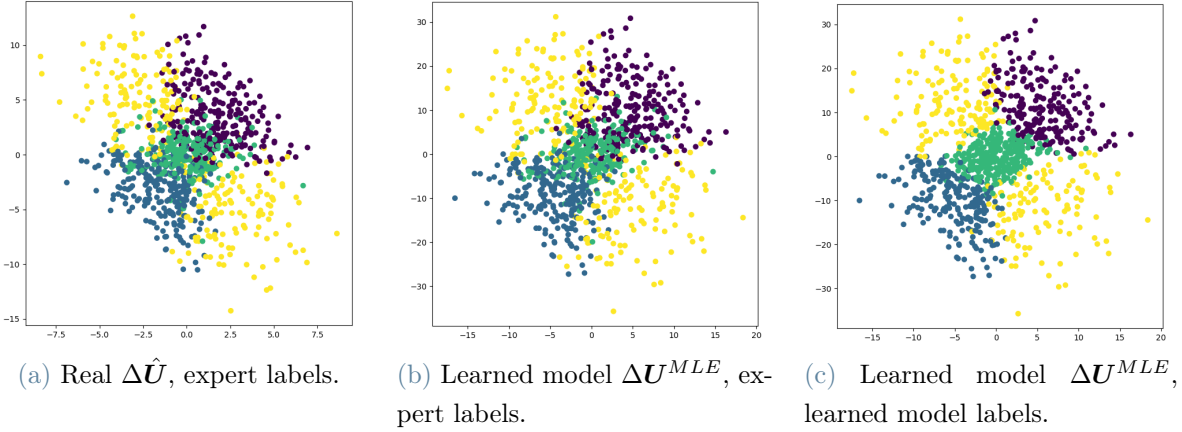


Figure 6.2: Test set pairs real and learned mappings for Model B in the $\Delta U_1/\Delta U_2$ space. Labels: preference (\succ), inverted preference (\prec), indifference (\asymp), incomparability (\parallel).

Model C The learned parameter matrix is:

$$\mathbf{W}_C^{MLE} \simeq \begin{pmatrix} 1.649 & 0.264 & -1.084 \\ 2.141 & 0.471 & -0.393 \end{pmatrix} \quad (6.4)$$

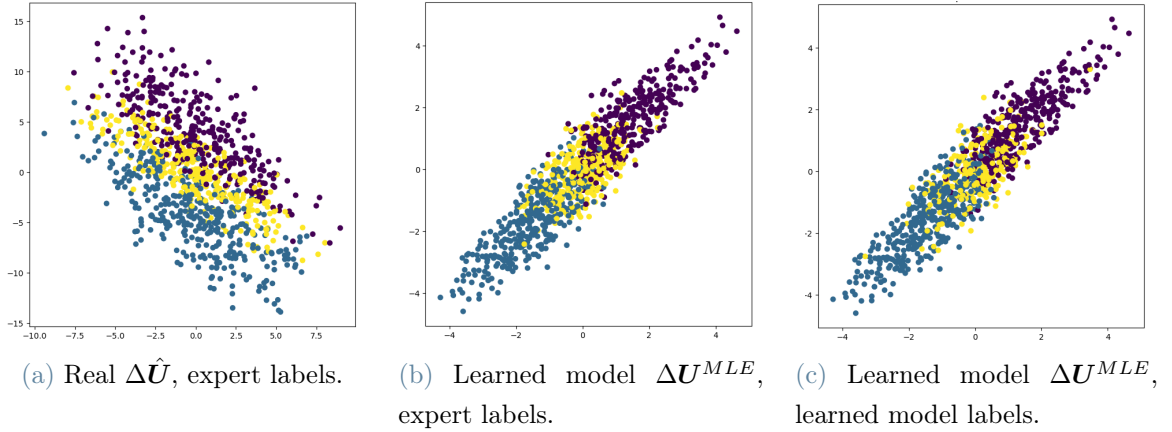


Figure 6.3: Test set pairs real and learned mappings for Model C in the $\Delta U_1/\Delta U_2$ space. Labels: preference (\succ), inverted preference (\prec), incomparability (\parallel).

The learned mapping for Model C, along with those obtained in other experimental settings using this model, reflect a broader pattern: it can be seen that the optimal point \mathbf{W}_C^{MLE} always tends to a matrix whose rows are all equal to each other. This is expected, since it can be shown that the gradient of $l(\mathbf{W})$ with respect to any row of the matrix is always the same, i.e.: $\frac{\partial l(\mathbf{W})}{\partial \mathbf{w}_i} = \frac{\partial l(\mathbf{W})}{\partial \mathbf{w}_j}, \forall i, j \in \llbracket k \rrbracket$.

A scatter plot of the test set pairs' utility values for Model C is depicted in Figure 6.3. It can be seen that, once the mapping has been learned, the decision boundaries are perpendicular to the standard diagonal in \mathbb{R}^d . Similar rows of the reconstructed \mathbf{W} give points that lie close to the standard diagonal, therefore the model is simply learning a mapping such that to distinguish strict preferences and a class placed near the origin.

For what concerns Model A and Model B, the scatter plots of the test set pairs' ΔU values are shown in Figure 6.1 (Model A) and Figure 6.2 (Model B). We note how, in both cases, the learned mapping is such that the majority of sampled labels correspond to the true labels. We can see how, for both Model A and Model B, the reconstructed \mathbf{W}^{MLE} is very close to the real $\hat{\mathbf{W}}$, apart from a scale factor. This behavior is not necessarily required, however, it is typically obtained for small d and k .

Different model for the expert, Jensen-Shannon divergence We analyzed all possible (9) cases of model class for the expert and model class for the learner. We have decided to use the Jensen-Shannon divergence [34] as a metric of distance between the distributions of the classes labeled by the expert and the learner. Results are presented in Table 6.1.

	Expert A	Expert B	Expert C
Learner A	0.0048	0.0072	0.0332
Learner B	0.0082	0.0043	0.0303
Learner C	0.0792	0.1347	0.0032

Table 6.1: Average Jensen-Shannon divergence for all combinations of learner/expert models (lower is better).

As can be seen in Table 6.1, Model A and Model B can learn their parameters such that they adapt to a real expert model chosen between Model A and Model B (in other words, these model are almost interchangeable). As for Model C, this table suggests, once again, its limitations: the high JS divergence values show that this model is not capable of learning decision surfaces that match the ones of Model A and Model B.

6.3. Multi-Objective Gridworld

Description

In the following, we evaluate our models on a gridworld environment with multiple, contrasting objectives. First, we formalize the environment and discuss the feature mapping ϕ and the trajectory generation process required for constructing a synthetic preference dataset. Then, we report and discuss the experimental results using the models introduced in Section 5. Finally, we evaluate the robustness of our methods in the cases of small datasets and incorrect experts.

We consider a square grid composed of G^2 cells, with each cell (state) s denoted as a vector $(x_s, y_s)^\top \in \{0, \dots, G - 1\}^2$ of its coordinates in the grid. An agent is able to move freely in this grid, starting from a specific cell (starting cell, denoted as s_S) by selecting, at each time step, an action between going left, right, up, down or staying in the current cell. Some cells have obstacles in them, which the agent cannot traverse; we denote the set of obstacle cells as \mathcal{S}_O . If the selected action is such that the agent would escape from the grid or arrive in an obstacle cell, the action is overwritten to stay in the current cell. Since there is no randomness in the transition model, we can define a function that, given a state and an action, returns the state in which the agent will deterministically be at the next timestep, i.e., the function $t : \mathcal{S} \times \mathcal{A} \rightarrow \mathcal{S}$. One cell acts as the goal cell, denoted as s_G . Note that reaching the goal state does not terminate the trajectory. Instead, the agent continues to act until H timesteps have elapsed.

We have defined several real scalar reward functions, that are only dependent on the state in which the agent would arrive by selecting a specific action in a specific cell. In mathematical terms, we have $R(s, a) = R(s')$, with $s' = t(s, a)$. The defined reward functions are:

$$\begin{aligned} R_G(s) &:= \begin{cases} 1, & s = s_G \\ -0.1, & \textit{otherwise} \end{cases} \\ R_R(s) &:= -\frac{1}{2}y_s \\ R_O(s) &:= \begin{cases} -1, & \exists s' \in \mathcal{S}_O : \|s - s'\|_\infty \leq 1 \\ 0, & \textit{otherwise} \end{cases} \end{aligned} \quad (6.5)$$

Informally, R_G gives a positive reward to the agent in the goal state s_G only, while penalizing it for every step taken outside of that cell (reward of -0.1); R_R penalizes the agent the more she stays in the lower rows of the gridworld; finally, R_O penalizes the agent every time she arrives in a cell that is contiguous with an obstacle.

Note that, in the general case, the various rewards could be in contrast with each other or not. We chose these specific rewards and the layout of the starting cell, goal cell and obstacles such that there is a trade-off between the different rewards, in order to introduce a considerable amount of incomparabilities in the dataset.

We can formalize the environment as a MDP\(\backslash\)R $\mathcal{M}_{GW} := \langle \mathcal{S}, \mathcal{A}, H, P, \mu_0 \rangle$, where:

- $\mathcal{S} = \{(x, y), x, y \in \{0, \dots, G - 1\}\}$
- $\mathcal{A} = \{\text{“up”}, \text{“down”}, \text{“left”}, \text{“right”}, \text{“stay”}\}$
- $P(s'|s, a) = \begin{cases} 1, & s' = t(s, a) \\ 0, & \textit{otherwise} \end{cases}$
- $\mu_0(s) = \begin{cases} 1, & s = s_S \\ 0, & \textit{otherwise} \end{cases}$

Given a trajectory $\tau = (s_t, a_t)_{t \in [H]}$, we consider the trajectory feature mapping $\phi : \mathcal{T} \rightarrow \mathbb{N}^{|\mathcal{S}|}$ where $\phi_i(\tau) := N(s_i, \tau)$, $i \in [|\mathcal{S}|]$, and $N(s_i, \tau)$ is the number of times the state s_i is visited in τ .

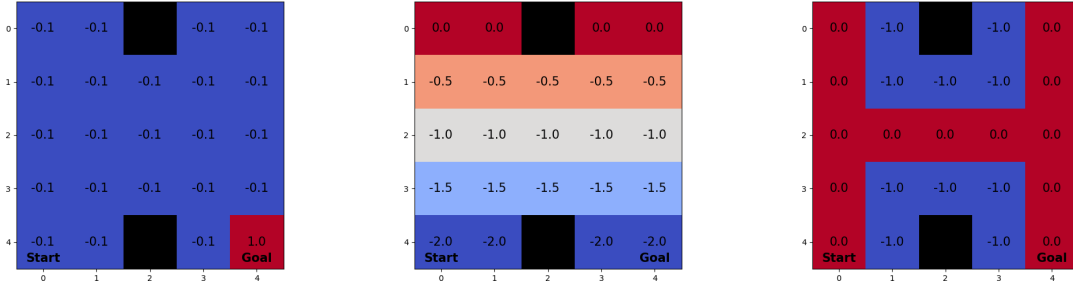
We consider a dataset of pre-collected trajectories obtained by playing a random policy, i.e., $\pi_{RND}(a|s) = \frac{1}{|\mathcal{A}|} \cdot \mathbf{1}_{|\mathcal{A}|}$, $\forall s \in \mathcal{S}, \forall a \in \mathcal{A}$.

The expert labels pairs of trajectories using a multi-dimensional real reward function \mathbf{R} ,

obtained juxtaposing some of the scalar reward functions defined in Equation 6.5 into a vector. In particular, we considered the following scenarios:

$$\begin{aligned}\mathbf{R}_R(s) &:= (R_G(s), R_R(s))^\top, \\ \mathbf{R}_O(s) &:= (R_G(s), R_O(s))^\top, \\ \mathbf{R}_B(s) &:= (R_G(s), R_R(s), R_O(s))^\top.\end{aligned}\tag{6.6}$$

In our experiments, we consider the case $G = 5$, $s_S = (0, 4)^\top$ (bottom left cell), $s_G = (4, 4)^\top$ (bottom right cell), $\mathcal{S}_O = \{(2, 0)^\top, (2, 4)^\top\}$. The real objective functions values over the grid are depicted in Figure 6.4. The datasets, where not specified, were collected by sampling $N_T = 300$ trajectories using π_{RND} , and querying the oracle to label each possible pair, i.e., $N = \binom{300}{2} = 44850$ pairs.



(a) $R_G(s)$ over the gridworld. (b) $R_R(s)$ over the gridworld. (c) $R_O(s)$ over the gridworld.

Figure 6.4: True reward functions over the 5×5 gridworld.

Results

To assess the variance of the found solution, each experiment was evaluated $N_{eval} = 10$ times, with different initial parameter estimates; the negative log-likelihood minimum, maximum, mean values and standard deviation over the different runs are shown. To qualitatively assess the found parameter estimate, we show the single components (rows) of the learned reward matrix \mathbf{W}^{MLE} , reshaped to obtain the shape of the grid (5×5), for one of the runs. This is reasonable because, given the choice of the feature vector ϕ , in which the i -th component is the number of times the state s_i is visited in the trajectory, the component $W_{d,i}^{MLE}$ encodes the relative importance of state s_i for the d -th component of the utility.

Bradley-Terry model As a preliminary experiment, the Bradley-Terry model (see Equation 2.17) was fitted to this setting. Clearly, in this case the learned utility is scalar ($U(\tau) = \mathbf{w}^{MLE\top} \phi(\tau)$). We employed the method used by [16, 27] for dealing with non-strict preferences: equally weighting indifferences over the preference classes (see Equation 3.1), and ignoring incomparabilities in the dataset. We note that Model A and Model B, when ignoring the incomparability class, become exactly equivalent to Model C and, therefore, to the Bradley-Terry with Ties model (Equation 3.2) in which we use a scalarization of the reward components with uniform weights. Therefore, we expect to learn a scalar utility function that is an average of the utility components.

Further experiments were carried on for Model A and Model B only, discarding Model C because inherently similar to the classic BT model. As already discussed in Section 6.2, Model C becomes useless in learning distinct reward dimensions as it simply tends to learn the same values for the d rows of \mathbf{W} .

Results in the case the expert model is Model A or Model B are shown in Figure 6.5 (real reward is \mathbf{R}_O) and Figure 6.6 (real reward is \mathbf{R}_B).

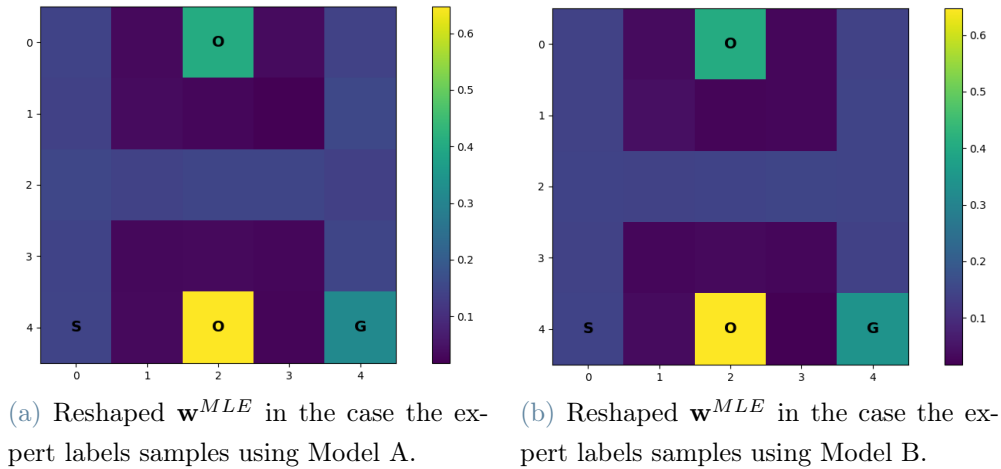


Figure 6.5: Bradley-Terry learned \mathbf{w}^{MLE} , true reward is \mathbf{R}_O .

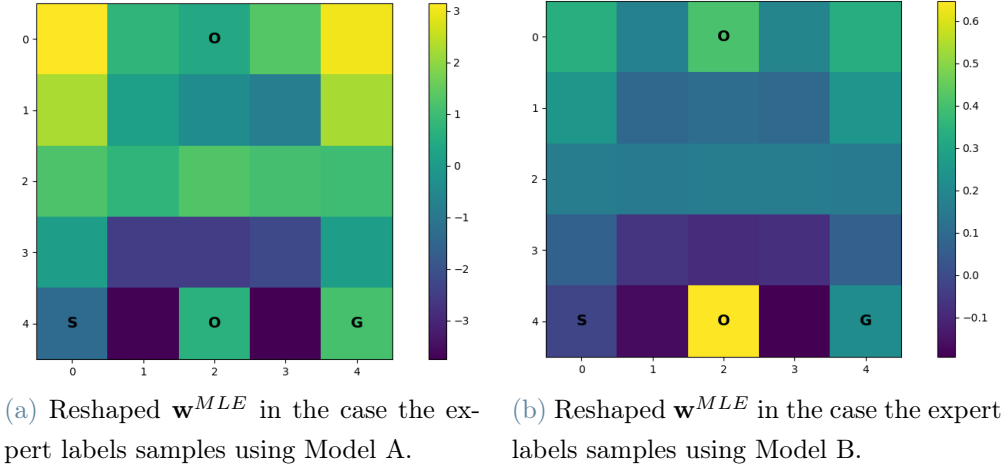


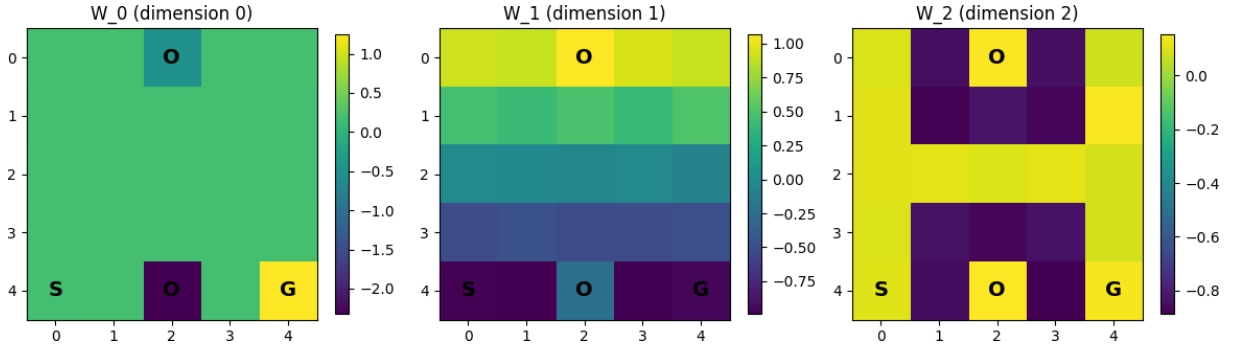
Figure 6.6: Bradley-Terry learned \mathbf{w}^{MLE} , true reward is \mathbf{R}_B .

$l(\mathbf{w}^{MLE})$	mean	std	min	max
BT, expert A (\mathbf{R}_O)	20261.764	0.0594	20261.669	20261.856
BT, expert B (\mathbf{R}_O)	19296.213	0.1208	19296.026	19296.345
BT, expert A (\mathbf{R}_B)	2730.011	0.0384	2729.98	2730.117
BT, expert B (\mathbf{R}_B)	3522.936	0.0190	3522.913	3522.974

Table 6.2: Negative log-likelihood statistics for the Bradley-Terry model.

We note that the standard deviation of the optimum values is very low over the re-runs; this is because the optimization problem is, in this case, convex, and therefore $l(\mathbf{w}^{MLE})$ always tends to the same unique minimum. Qualitatively, we notice that, as previously hypothesized, the learned reward vector tends to the average of the reward dimensions.

Model B One learned \mathbf{W}_B^{MLE} , in the case the real reward is \mathbf{R}_B , is shown in Figure 6.7.

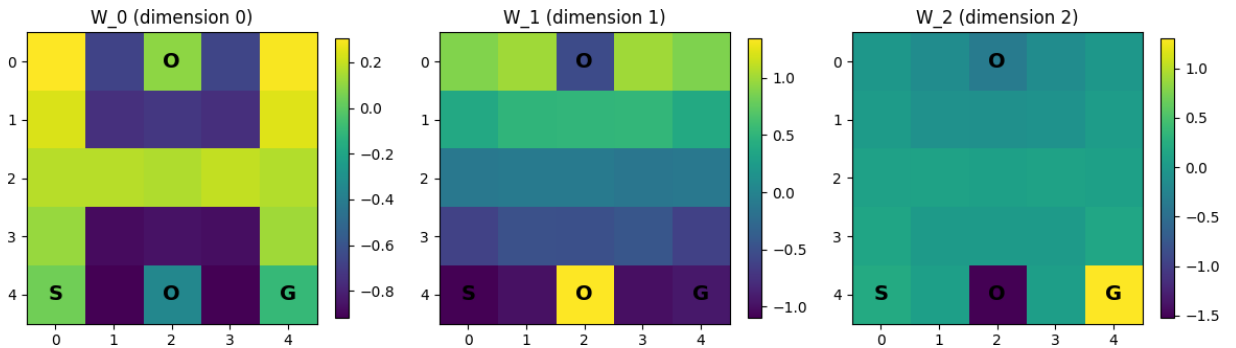
Figure 6.7: Learned \mathbf{W}_B^{MLE} , true reward is \mathbf{R}_B .

$l(\mathbf{W}^{MLE})$	mean	std	min	max
learner B, expert B (\mathbf{R}_B)	13277.834	23.964	13264.598	13345.026

Table 6.3: Negative log-likelihood statistics for Model B.

Even in the case of Model B, the learned parameter estimates over the various re-runs are very close to each other (see Table 6.3) and all tend to a qualitatively correct reward function (whose example is depicted in Figure 6.7). This is due to the fact that, even if the negative log-likelihood function is not convex, if the number of incomparable pairs is limited with respect to the comparable ones, it displays a convex-like profile, with smooth curvature and no pronounced local irregularities.

Model A For what concerns Model A, one learned \mathbf{W}_A^{MLE} , in the case the real reward is \mathbf{R}_B , is shown in Figure 6.8.

Figure 6.8: Learned \mathbf{W}_A^{MLE} , true reward is \mathbf{R}_B .

$l(\mathbf{W}^{MLE})$	mean	std	min	max
learner A, expert A (R_B)	5202.656	1.082	5201.780	5204.739

Table 6.4: Negative log-likelihood statistics for Model A.

The same remarks as in the case of Model B apply here. We note that in this case, however, we obtained, on average, a slightly noisier estimate of $\hat{\mathbf{W}}$ as optimum point. This can be visually seen in Figure 6.8, in which the reward components are not separated as in the case of Model B. This could be due to the fact that the negative log-likelihood function presents more substantial local irregularities than the one of Model B.

In Table 6.5, we show the average classification metrics over the re-runs. These act as a baseline, to be compared with the robustness studies ones.

	Precision	Recall	F1-Score
Model A	0.932	0.933	0.932
Model B	0.904	0.904	0.904

Table 6.5: Classification metrics for Models A and B (average over re-runs).

Small datasets This series of experiments aims at studying the learned parameter estimate \mathbf{W}^{MLE} using, as dataset, an incomplete graph of comparisons between trajectories. In Figure 6.11, we show results for 100, 500, 1000, 2000 and 5000 trajectory pairs, using Model B (similar results are obtained for Model A). In Table 6.6, we present the classification metrics at increasing values of N

	Precision	Recall	F1-Score
N = 500	0.819	0.366	0.314
N = 1000	0.827	0.327	0.280
N = 2000	0.856	0.582	0.609
N = 5000	0.863	0.650	0.676

Table 6.6: Classification metrics at increasing values of N. (average over re-runs)

We note that the reconstructed reward matrix is qualitatively similar to the real $\hat{\mathbf{W}}$ starting from datasets having size of few thousands samples (this is supported by the

sudden increase in classification metrics). We note that the reconstruction of the weight matrix gets progressively better and less noisy with the increasing number of samples N . The model displays a particular difficulty in reconstructing the R_R component; this aligns with the intuition that more uniform rewards, and the ones that assume values in a smaller set are easier to learn with respect to uneven and many-valued ones (R_G and R_O can assume only 2 values over the grid, while R_R takes values in a set of cardinality 5).

Studying the robustness with respect to the underlying expert model We considered two alternative scenarios, in which the expert model does not correspond with a previously defined model. In particular we define the following expert models:

- $(1 - \epsilon)$ -correct expert: chosen an expert model class γ , given (τ_1, τ_2) with probability $(1 - \epsilon)$ it outputs the label $y_{ij} \sim \gamma(\Delta \mathbf{U}(\tau_1, \tau_2))$, and with probability ϵ , it outputs a random (uniformly sampled) class in $\mathcal{Y} \setminus \{y_{ij}\}$.
- $(1 - \epsilon)$ -argmax expert: chosen an expert model class γ , given (τ_1, τ_2) with probability $(1 - \epsilon)$ it outputs the label $y_{ij} = \arg \max_y \{\gamma_y(\Delta \mathbf{U}(\tau_1, \tau_2)), i \in \mathcal{Y}\}$, and with probability ϵ , it outputs a random (uniformly sampled) class in $\mathcal{Y} \setminus \{y_{ij}\}$.

We display results for Model B (again, similar results are achieved by Model A). In Figure 6.12, learned parameter matrices for increasing values of ϵ are shown in the case of $(1 - \epsilon)$ -correct expert, in Figure 6.13 we show the case of $(1 - \epsilon)$ -argmax expert.

We note how, in general, increasing the value of ϵ increases the noise in the reconstructed \mathbf{W}^{MLE} with respect to the real matrix $\hat{\mathbf{W}}$; this behavior is expected, since the expert model progressively deviates from the model employed by the learner, increasingly adding noise in the dataset's labels.

In the case of the $(1 - \epsilon)$ -correct expert, the performances (in terms of estimate \mathbf{W}^{MLE}) degrade more slowly with respect to the case of the $(1 - \epsilon)$ -argmax expert. This can be seen, for example, by looking at the many-valued reward R_R , which, for $\epsilon = 0.001$ is still well-captured by the model in the case of $(1 - \epsilon)$ -correct expert, while it's already very noisy for the $(1 - \epsilon)$ -argmax expert model. This is an outcome we somehow expect, since the $(1 - \epsilon)$ -argmax expert model is the most pessimistic and far from the learner's model between the two (due to both the use of the *argmax* operator and the ϵ -probability behavior).

	Precision	Recall	F1-Score
$\epsilon = 0.001$	0.749	0.739	0.744
$\epsilon = 0.01$	0.786	0.782	0.784
$\epsilon = 0.1$	0.702	0.710	0.706
$\epsilon = 0.2$	0.651	0.630	0.642

Table 6.7: Classification metrics at increasing values of ϵ . ($(1 - \epsilon)$ -correct expert, average over re-runs).

	Precision	Recall	F1-Score
$\epsilon = 0$	0.968	0.962	0.964
$\epsilon = 0.001$	0.959	0.950	0.954
$\epsilon = 0.01$	0.911	0.890	0.900
$\epsilon = 0.1$	0.729	0.749	0.739

Table 6.8: Classification metrics at increasing values of ϵ . ($(1 - \epsilon)$ -argmax expert, average over re-runs).

We show average classification metrics in both cases in Table 6.7 ($(1 - \epsilon)$ -correct expert) and Table 6.8 ($(1 - \epsilon)$ -argmax expert). We note, in both cases, a substantial drop in all classification metrics for increasing values of ϵ .

Mini-batch gradient descent As introduced in Section 6.1, we developed implementations of Model A and Model B in PyTorch, using mini-batch gradient descent with the Adam optimizer. We note how even these implementations reach parameter estimates \mathbf{W}^{MLE} that are qualitatively very close to the true parameter $\hat{\mathbf{W}}$ (see Figure 6.10 and Figure 6.9). Even if these results are not as good as the ones obtained using the L-BFGS optimization algorithm, Furthermore, we noticed a substantial speed-up in the optimization process, reaching around x10 decrease in runtime, for both Model A and Model B.

	Precision	Recall	F1-Score
Model A (PyTorch)	0.810	0.760	0.687
Model B (PyTorch)	0.756	0.761	0.758

Table 6.9: Classification metrics for Models A and B (PyTorch implementation, average over the re-runs).

We note a slight worsening in all classification metrics (Table 6.9). This is due to the fact that the mini-batch gradient descent converges to a noisier estimate of \mathbf{W} with respect to BFGS.

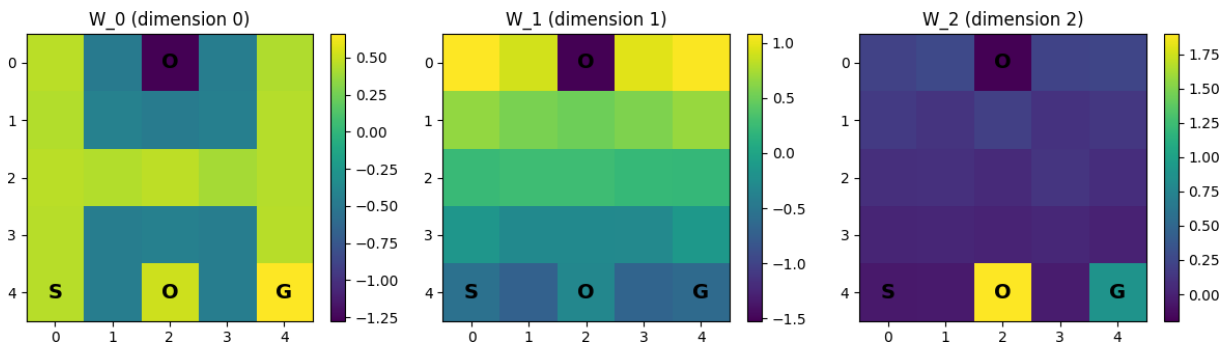


Figure 6.9: \mathbf{W}_A^{MLE} obtained with mini-batch gradient descent.

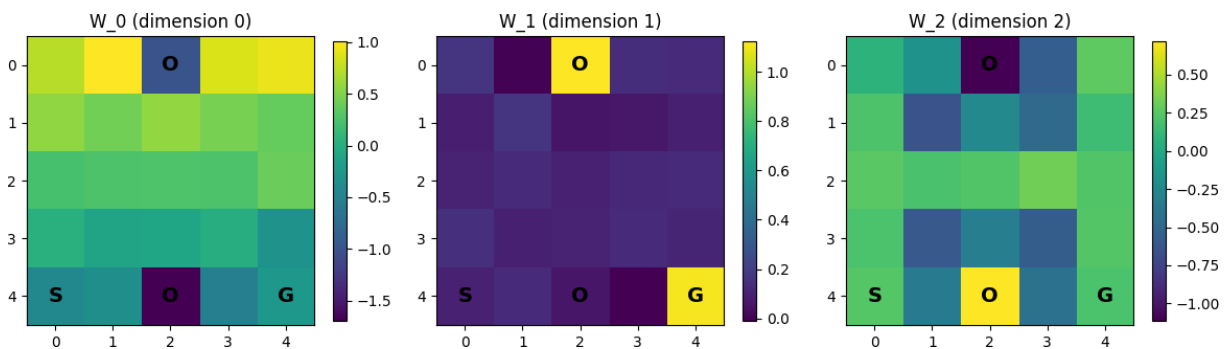


Figure 6.10: \mathbf{W}_B^{MLE} obtained with mini-batch gradient descent.

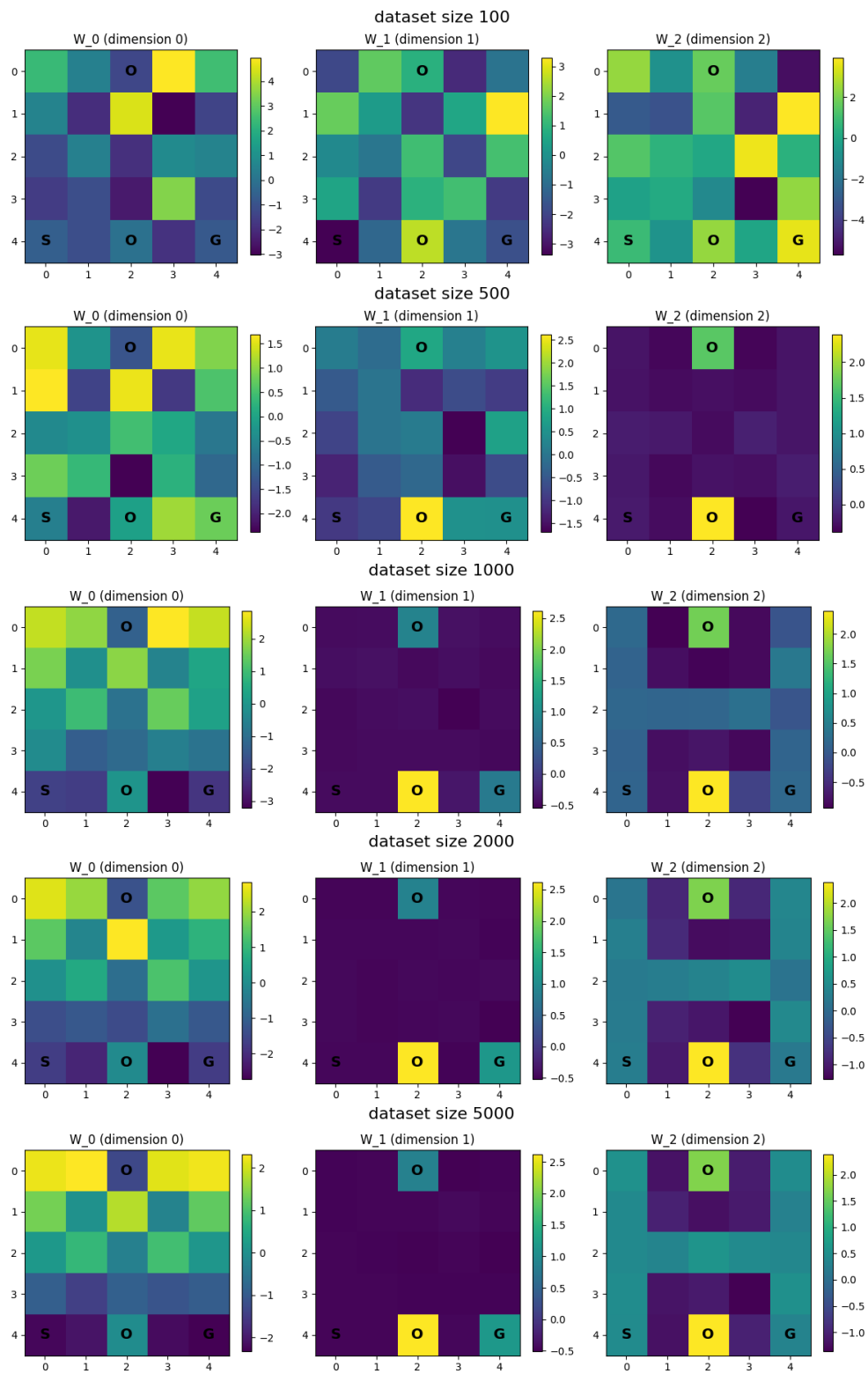


Figure 6.11: Learned reward matrices for increasing dataset size (Model B, true reward is R_B).

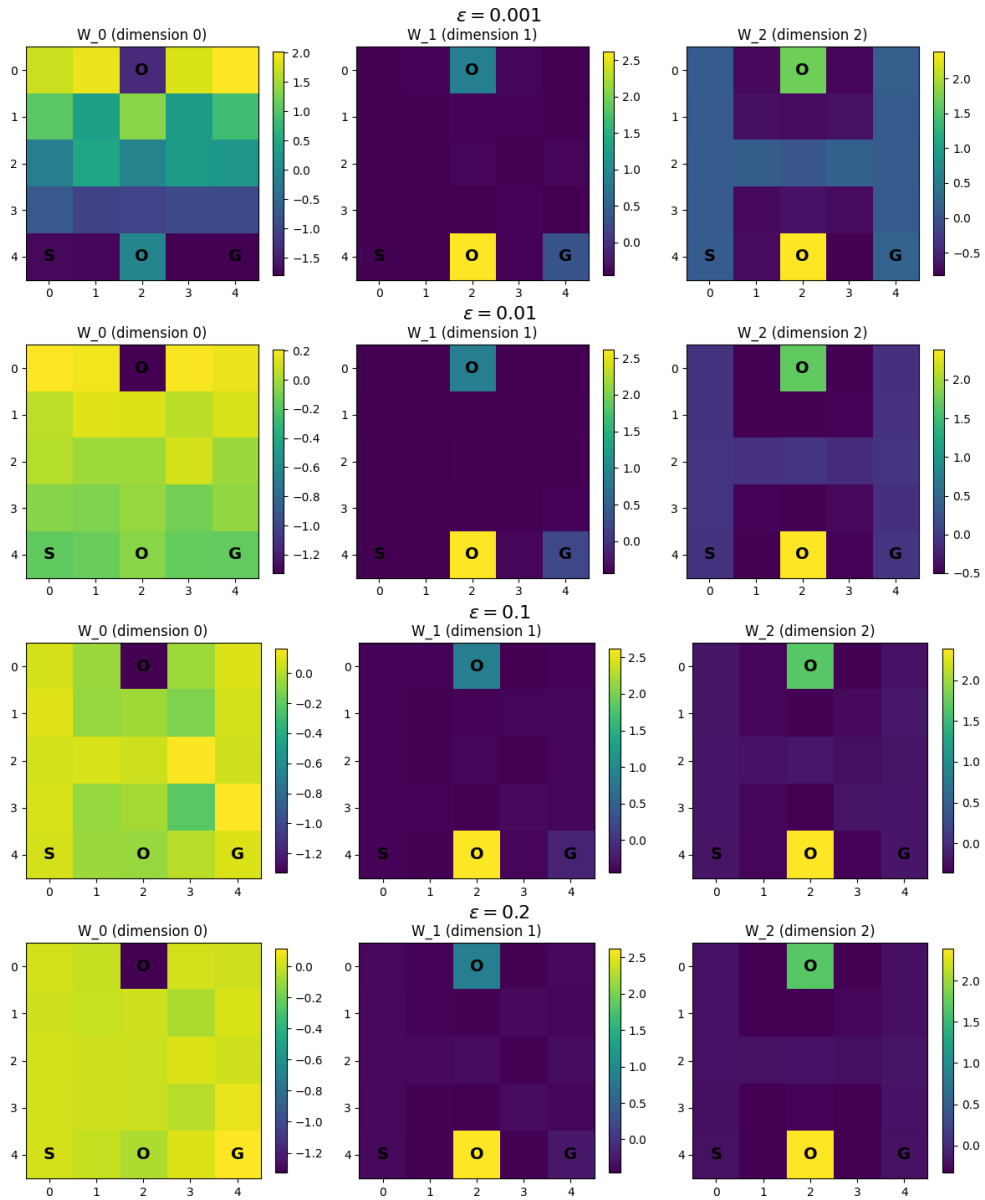


Figure 6.12: $(1 - \epsilon)$ -correct expert, increasing values of ϵ (Model B, true reward is \mathbf{R}_B).

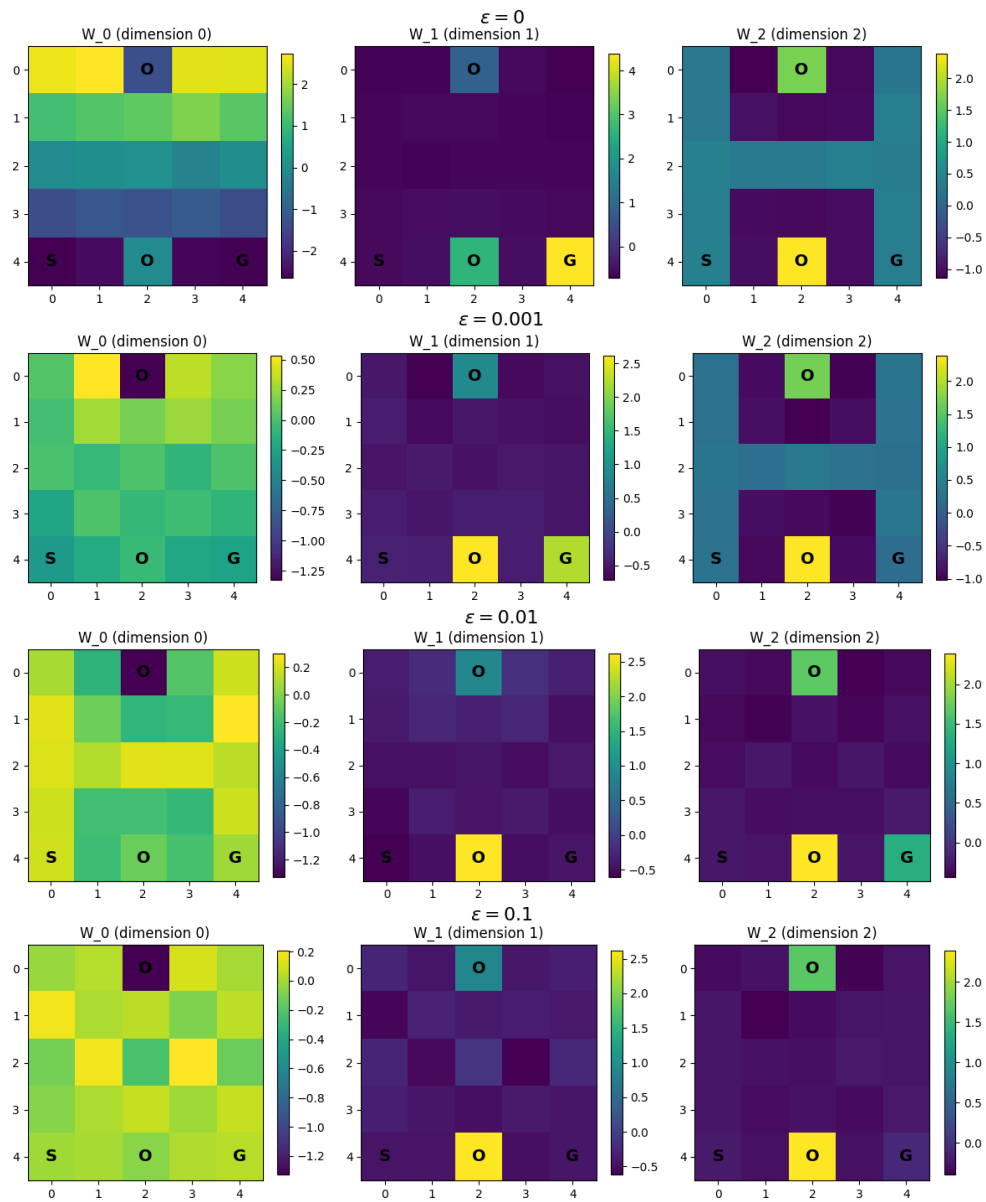


Figure 6.13: $(1 - \epsilon)$ -argmax expert, increasing values of ϵ (Model B, true reward is R_B).

7 | Conclusions

This thesis addressed the problem of learning multi-dimensional reward models from an expert-submitted feedback consisting of preferences, indifferences and incomparabilities between pairs of agent trajectories. As already noted in previous works, this setting is tied to the expert utilizing an underlying reward that is multi-dimensional; this is a very frequent case in real applications, such as autonomous driving, LLM alignment, water reservoir operations, etc). However, until now, learning in the presence of this particular feedback type has been avoided by the literature, by simply discarding the incomparable pairs. We formalized this setting by slightly modifying the classical PbRL settings and introduced the rational multi-objective expert model, a set of properties that the probabilistic model of the expert must adhere to be coherent in the evaluation of trajectories with respect to multiple underlying objectives. We then proceeded by stating and proving an important negative result, i.e., the non-convexity, in the general case, of the negative log-likelihood of any model complying with such properties. We therefore started illustrating various possible model choices: Model A and Model B, that respect completely the desiderata on the problem, while having non-convex losses and Model C, a model that is only partially able to represent incomparabilities but that benefits from a convex loss. Particular attention, throughout the whole process, has been placed on defining models that, in the classical one-dimensional case, reduced to the ubiquitous Bradley-Terry model. Finally, we tested our models throughout diverse experiments, validating the optimization procedure and conducting studies on datasets having particularly limited cardinality and verifying the robustness of the model with respect to different, imperfect expert models.

7.1. Future Works

Learning a set of policies The final objective of any classical PbRL algorithm is to find an optimal policy π^* , and the reward model learning phase is a step in a bigger, more complex process. Even in the problem setting we defined, the final goal should always be the one of controlling the environment by learning a suitable policy, or, more precisely, a set of non-dominated policies since we are dealing with a multi-objective problem.

An interesting research direction would be the development of such end-to-end learning algorithm, by (a) learning the multi-dimensional reward model from data and (b) learning a set of non dominated policies on the MOMDP defined using such reward. In the case of an online learning setup (i.e., the agent can actively interact with the environment and query the expert), ulterior issues have to be addressed, such as:

- The dimensionality of the reward model should be increased with the number of incomparabilities in the dataset ζ . More specifically, this could be done by iteratively updating a sample-based estimate of the width of the preference relation corresponding to the dataset.
- The preference query generation problem, i.e., selecting the most informative trajectory pair to be submitted for feedback to the expert. In this specific setting, the learner should take into consideration queries that make it possible to discover new objectives, while also exploring the already known ones.

Evaluation on real datasets Systematically overlooking the possibility of incomparabilities in the dataset also implies that there exist no datasets for this exact problem setting. In particular, to the best of our knowledge, no public dataset of preferences comprising incomparabilities exists (across any application field). Being able to evaluate the goodness of expert models with respect to real preference data comprising incomparabilities would be of paramount importance to understand if such models are able to mimic the decision making strategies of real human labelers.

Sample complexity Sample complexity is a fundamental concept in RL (and ML in general). The sample complexity analysis of an RL algorithm is focused on obtaining an asymptotic upper bound on the regret of such algorithm, as a function of the observed samples. Several works tackled this problem in the PbRL literature [40, 49, 69]. One possible work direction is studying the sample complexity for an algorithm learning a set of Pareto-optimal policies from human preferences with incomparabilities, in which a suitable multi-dimensional reward model is utilized.

Bibliography

- [1] P. Abbeel and A. Y. Ng. Apprenticeship learning via inverse reinforcement learning. In *Proceedings of the Twenty-First International Conference on Machine Learning, ICML '04*, page 1, New York, NY, USA, 2004. Association for Computing Machinery. ISBN 1581138385. doi: 10.1145/1015330.1015430. URL <https://doi.org/10.1145/1015330.1015430>.
- [2] R. Akrouf, M. Schoenauer, and M. Sebag. Preference-based policy learning. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 12–27. Springer, 2011.
- [3] D. Amodei, C. Olah, J. Steinhardt, P. Christiano, J. Schulman, and D. Mané. Concrete problems in ai safety, 2016. URL <https://arxiv.org/abs/1606.06565>.
- [4] M. G. Azar, M. Rowland, B. Piot, D. Guo, D. Calandriello, M. Valko, and R. Munos. A general theoretical paradigm to understand learning from human preferences, 2023. URL <https://arxiv.org/abs/2310.12036>.
- [5] L. Barrett and S. Narayanan. Learning all optimal policies with multiple criteria. In *Proceedings of the 25th International Conference on Machine Learning, ICML '08*, page 41–47, New York, NY, USA, 2008. Association for Computing Machinery. ISBN 9781605582054. doi: 10.1145/1390156.1390162. URL <https://doi.org/10.1145/1390156.1390162>.
- [6] R. Bellman. Dynamic programming. *Science*, 153(3731):34–37, 1966. ISSN 00368075, 10959203. URL <http://www.jstor.org/stable/1719695>.
- [7] C. Bishop. *Pattern Recognition and Machine Learning*. Springer, January 2006. URL <https://www.microsoft.com/en-us/research/publication/pattern-recognition-machine-learning/>.
- [8] R. Bommasani. On the opportunities and risks of foundation models. *arXiv preprint arXiv:2108.07258*, 2021.

- [9] S. P. Boyd and L. Vandenberghe. *Convex optimization*. Cambridge university press, 2004.
- [10] R. A. Bradley and M. E. Terry. Rank analysis of incomplete block designs: I. the method of paired comparisons. *Biometrika*, 39(3/4):324–345, 1952. ISSN 00063444, 14643510. URL <http://www.jstor.org/stable/2334029>.
- [11] R. Busa-Fekete, B. Szörényi, P. Weng, W. Cheng, and E. Hüllermeier. Preference-based evolutionary direct policy search. In *ICRA Workshop on autonomous learning*, volume 2, 2013.
- [12] R. Busa-Fekete, B. Szörényi, P. Weng, W. Cheng, and E. Hüllermeier. Preference-based reinforcement learning: evolutionary direct policy search using a preference-based racing algorithm. *Machine learning*, 97(3):327–351, 2014.
- [13] R. H. Byrd, P. Lu, J. Nocedal, and C. Zhu. A limited memory algorithm for bound constrained optimization. *SIAM Journal on scientific computing*, 16(5):1190–1208, 1995.
- [14] P. Chalermsook, B. Laekhanukit, and D. Nanongkai. *Graph Products Revisited: Tight Approximation Hardness of Induced Matching, Poset Dimension and More*, pages 1557–1576. doi: 10.1137/1.9781611973105.112. URL <https://epubs.siam.org/doi/abs/10.1137/1.9781611973105.112>.
- [15] W.-L. Chiang, L. Zheng, Y. Sheng, A. N. Angelopoulos, T. Li, D. Li, B. Zhu, H. Zhang, M. I. Jordan, J. E. Gonzalez, and I. Stoica. Chatbot arena: an open platform for evaluating llms by human preference. In *Proceedings of the 41st International Conference on Machine Learning*, ICML’24. JMLR.org, 2024.
- [16] P. F. Christiano, J. Leike, T. Brown, M. Martic, S. Legg, and D. Amodei. Deep reinforcement learning from human preferences. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper_files/paper/2017/file/d5e2c0adad503c91f91df240d0cd4e49-Paper.pdf.
- [17] R. R. Davidson. On extending the bradley-terry model to accommodate ties in paired comparison experiments. *Journal of the American Statistical Association*, 65(329): 317–328, 1970. ISSN 01621459, 1537274X. URL <http://www.jstor.org/stable/2283595>.
- [18] R. P. Dilworth. A decomposition theorem for partially ordered sets. *Annals of*

- Mathematics*, 51(1):161–166, 1950. ISSN 0003486X, 19398980. URL <http://www.jstor.org/stable/1969503>.
- [19] S. Drago, M. Mussi, A. M. Metelli, et al. Towards theoretical understanding of sequential decision making with preference feedback. In *42nd International Conference on Machine Learning, ICML 2025*, pages 1–16, 2025.
- [20] B. Dushnik and E. W. Miller. Partially ordered sets. *American Journal of Mathematics*, 63(3):600–610, 1941. ISSN 00029327, 10806377. URL <http://www.jstor.org/stable/2371374>.
- [21] A. Elo. The rating of chessplayers, past and present 2nd ed. *New York: Arco*, 1986.
- [22] Ö. Evren and E. A. Ok. On the multi-utility representation of preference relations. *Journal of Mathematical Economics*, 47(4-5):554–563, 2011.
- [23] S. Felsner, I. Mustata, and M. Pergel. The complexity of the partial order dimension problem - closing the gap, 2016. URL <https://arxiv.org/abs/1501.01147>.
- [24] J. Fürnkranz, E. Hüllermeier, W. Cheng, and S.-H. Park. Preference-based reinforcement learning: a formal framework and a policy iteration algorithm. *Machine learning*, 89(1):123–156, 2012.
- [25] C. F. Hayes, R. Rădulescu, E. Bargiacchi, J. Källström, M. Macfarlane, M. Reymond, T. Verstraeten, L. M. Zintgraf, R. Dazeley, F. Heintz, E. Howley, A. A. Irissappane, P. Mannion, A. Nowé, G. Ramos, M. Restelli, P. Vamplew, and D. M. Roijers. A practical guide to multi-objective reinforcement learning and planning. *Autonomous Agents and Multi-Agent Systems*, 36(1), Apr. 2022. ISSN 1573-7454. doi: 10.1007/s10458-022-09552-y. URL <http://dx.doi.org/10.1007/s10458-022-09552-y>.
- [26] X. He and C. Lv. Toward personalized decision making for autonomous vehicles: a constrained multi-objective reinforcement learning technique. *Transportation research part C: emerging technologies*, 156:104352, 2023.
- [27] B. Ibarz, J. Leike, T. Pohlen, G. Irving, S. Legg, and D. Amodei. Reward learning from human preferences and demonstrations in atari. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018. URL https://proceedings.neurips.cc/paper_files/paper/2018/file/8cbe9ce23f42628c98f80fa0fac8b19a-Paper.pdf.
- [28] P. Jain, P. Kar, et al. Non-convex optimization for machine learning. *Foundations and Trends® in Machine Learning*, 10(3-4):142–363, 2017.

- [29] H. J. Jeon, S. Milli, and A. Dragan. Reward-rational (implicit) choice: A unifying formalism for reward learning. In H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 4415–4426. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/2f10c1578a0706e06b6d7db6f0b4a6af-Paper.pdf.
- [30] T. Kaufmann, P. Weng, V. Bengs, and E. Hüllermeier. A survey of reinforcement learning from human feedback, 2024. URL <https://arxiv.org/abs/2312.14925>.
- [31] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization, 2017. URL <https://arxiv.org/abs/1412.6980>.
- [32] J. Kober, J. A. Bagnell, and J. Peters. Reinforcement learning in robotics: A survey. *The International Journal of Robotics Research*, 32(11):1238–1274, 2013.
- [33] S. Kullback and R. A. Leibler. On information and sufficiency. *The annals of mathematical statistics*, 22(1):79–86, 1951.
- [34] J. Lin. Divergence measures based on the shannon entropy. *IEEE Transactions on Information Theory*, 37(1):145–151, 1991. doi: 10.1109/18.61115.
- [35] J. Liu, D. Ge, and R. Zhu. Reward learning from preference with ties. *arXiv preprint arXiv:2410.05328*, 2024.
- [36] J. Liu, G. Liu, J. Liang, Z. Yuan, X. Liu, M. Zheng, X. Wu, Q. Wang, W. Qin, M. Xia, et al. Improving video generation with human feedback. *arXiv preprint arXiv:2501.13918*, 2025.
- [37] N. Mu, Y. Luan, and Q.-S. Jia. Preference-based multi-objective reinforcement learning. *IEEE Transactions on Automation Science and Engineering*, 22:18737–18749, 2025. doi: 10.1109/TASE.2025.3589271.
- [38] A. Y. Ng, D. Harada, and S. Russell. Policy invariance under reward transformations: Theory and application to reward shaping. In *Icml*, volume 99, pages 278–287. Citeseer, 1999.
- [39] J. Nocedal and S. J. Wright. *Numerical optimization*. Springer, 2006.
- [40] E. Novoseller, Y. Wei, Y. Sui, Y. Yue, and J. Burdick. Dueling posterior sampling for preference-based reinforcement learning. In *Conference on Uncertainty in Artificial Intelligence*, pages 1029–1038. PMLR, 2020.

- [41] E. A. Ok et al. Utility representation of an incomplete preference relation. *Journal of Economic Theory*, 104(2):429–449, 2002.
- [42] L. Ouyang, J. Wu, X. Jiang, D. Almeida, C. Wainwright, P. Mishkin, C. Zhang, S. Agarwal, K. Slama, A. Ray, et al. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744, 2022.
- [43] A. Pan, K. Bhatia, and J. Steinhardt. The effects of reward misspecification: Mapping and mitigating misaligned models. *arXiv preprint arXiv:2201.03544*, 2022.
- [44] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper_files/paper/2019/file/bdbca288fee7f92f2bfa9f7012727740-Paper.pdf.
- [45] R. L. Plackett. The analysis of permutations. *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, 24(2):193–202, 1975. ISSN 00359254, 14679876. URL <http://www.jstor.org/stable/2346567>.
- [46] M. L. Puterman. Markov decision processes. In *Stochastic Models*, volume 2 of *Handbooks in Operations Research and Management Science*, pages 331–434. Elsevier, 1990. doi: [https://doi.org/10.1016/S0927-0507\(05\)80172-0](https://doi.org/10.1016/S0927-0507(05)80172-0). URL <https://www.sciencedirect.com/science/article/pii/S0927050705801720>.
- [47] R. Rafailov, A. Sharma, E. Mitchell, S. Ermon, C. D. Manning, and C. Finn. Direct preference optimization: Your language model is secretly a reward model, 2024. URL <https://arxiv.org/abs/2305.18290>.
- [48] P. V. Rao and L. L. Kupper. Ties in paired-comparison experiments: A generalization of the bradley-terry model. *Journal of the American Statistical Association*, 62(317):194–204, 1967. doi: 10.1080/01621459.1967.10482901. URL <https://www.tandfonline.com/doi/abs/10.1080/01621459.1967.10482901>.
- [49] A. Saha, A. Pacchiano, and J. Lee. Dueling rl: Reinforcement learning with trajectory preferences. In *International conference on artificial intelligence and statistics*, pages 6263–6289. PMLR, 2023.

- [50] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, et al. Mastering the game of go with deep neural networks and tree search. *nature*, 529(7587):484–489, 2016.
- [51] D. Silver, S. Singh, D. Precup, and R. S. Sutton. Reward is enough. *Artificial intelligence*, 299:103535, 2021.
- [52] J. Skalse, N. Howe, D. Krasheninnikov, and D. Krueger. Defining and characterizing reward gaming. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, editors, *Advances in Neural Information Processing Systems*, volume 35, pages 9460–9471. Curran Associates, Inc., 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/3d719fee332caa23d5038b8a90e81796-Paper-Conference.pdf.
- [53] S. M. Stigler. Citation patterns in the journals of statistics and probability. *Statistical Science*, 9(1):94–108, 1994. ISSN 08834237. URL <http://www.jstor.org/stable/2246292>.
- [54] H. Sugiyama, T. Meguro, and Y. Minami. Preference-learning based inverse reinforcement learning for dialog control. In *INTERSPEECH*, volume 12, pages 222–225, 2012.
- [55] R. S. Sutton and A. G. Barto. *Reinforcement Learning: An Introduction*. A Bradford Book, Cambridge, MA, USA, 2018. ISBN 0262039249.
- [56] G. Tesauro et al. Temporal difference learning and td-gammon. *Communications of the ACM*, 38(3):58–68, 1995.
- [57] K. E. Train. *Discrete choice methods with simulation*. Cambridge university press, 2009.
- [58] W. Trotter. *Combinatorics and Partially Ordered Sets: Dimension Theory*. Johns Hopkins Studies in the Mathematical Sciences. Johns Hopkins University Press, 1992. ISBN 9780801844256. URL <https://books.google.it/books?id=oe-EAAAAIAAJ>.
- [59] H.-H. Tseng, Y. Luo, S. Cui, J.-T. Chien, R. K. Ten Haken, and I. E. Naqa. Deep reinforcement learning for automated radiation adaptation in lung cancer. *Medical physics*, 44(12):6690–6705, 2017.
- [60] P. Vamplew, R. Dazeley, A. Berry, R. Issabekov, and E. Dekker. Empirical evaluation methods for multiobjective reinforcement learning algorithms. *Machine learning*, 84(1):51–80, 2011.

- [61] P. Vamplew, R. Issabekov, R. Dazeley, and C. Foale. Reinforcement learning of pareto-optimal multiobjective policies using steering. In *Australasian Joint Conference on Artificial Intelligence*, pages 596–608. Springer, 2015.
- [62] P. Vamplew, B. J. Smith, J. Källström, G. Ramos, R. Rădulescu, D. M. Roijers, C. F. Hayes, F. Heintz, P. Mannion, P. J. Libin, et al. Scalar reward is not enough: A response to silver, singh, precup and sutton (2021). *Autonomous Agents and Multi-Agent Systems*, 36(2):41, 2022.
- [63] J. von Neumann, O. Morgenstern, and A. Rubinstein. *Theory of Games and Economic Behavior (60th Anniversary Commemorative Edition)*. Princeton University Press, 1944. ISBN 9780691130613. URL <http://www.jstor.org/stable/j.ctt1r2gkx>.
- [64] C. J. Watkins and P. Dayan. Q-learning. *Machine learning*, 8(3):279–292, 1992.
- [65] A. Wilson, A. Fern, and P. Tadepalli. A bayesian approach for policy learning from trajectory preference queries. In *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 1*, NIPS’12, page 1133–1141, Red Hook, NY, USA, 2012. Curran Associates Inc.
- [66] C. Wirth and J. Fürnkranz. A policy iteration algorithm for learning from preference-based feedback. In *International Symposium on Intelligent Data Analysis*, pages 427–437. Springer, 2013.
- [67] C. Wirth, J. Fürnkranz, and G. Neumann. Model-free preference-based reinforcement learning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 30, 2016.
- [68] C. Wirth, R. Akrouf, G. Neumann, and J. Fürnkranz. A survey of preference-based reinforcement learning methods. *Journal of Machine Learning Research*, 18(136): 1–46, 2017. URL <http://jmlr.org/papers/v18/16-634.html>.
- [69] Y. Xu, R. Wang, L. Yang, A. Singh, and A. Dubrawski. Preference-based reinforcement learning with finite-time guarantees. *Advances in Neural Information Processing Systems*, 33:18784–18794, 2020.
- [70] M. Yannakakis. The complexity of the partial order dimension problem. *SIAM Journal on Algebraic Discrete Methods*, 3(3):351–358, 1982. doi: 10.1137/0603036. URL <https://doi.org/10.1137/0603036>.
- [71] E. Yurtsever, J. Lambert, A. Carballo, and K. Takeda. A survey of autonomous

- driving: Common practices and emerging technologies. *IEEE access*, 8:58443–58469, 2020.
- [72] Y. Zhao, M. R. Kosorok, and D. Zeng. Reinforcement learning design for cancer clinical trials. *Statistics in medicine*, 28(26):3294–3315, 2009.
- [73] Y. Zhao, R. Joshi, T. Liu, M. Khalman, M. Saleh, and P. J. Liu. Slic-hf: Sequence likelihood calibration with human feedback, 2023. URL <https://arxiv.org/abs/2305.10425>.

List of Figures

2.1	Examples of order relations	7
2.2	Example of partial order, antichain and realizer.	8
2.3	Sketch of the interaction protocol between an RL agent and the environment.	10
5.1	Representation of the values f_{\parallel} and f_{\succ} on the $(\Delta U_1, \Delta U_2)$ plane.	38
5.2	Visualization of desired preference <i>argmax</i> area	39
5.3	$\arg \max_y \gamma_y((\Delta U_1, \Delta U_2)^\top)$ in the case $K = 1, \alpha = 0$ (Model A).	41
5.4	Non convexity of negative log-likelihood of Model A.	42
5.5	Plot of $f_{\parallel}(\Delta \mathbf{U}) = \text{std}(\Delta \mathbf{U})$ for $d = 2$	46
5.6	Output of Model C along the standard diagonal ($\alpha = 1$).	47
5.7	$\arg \max_y \gamma_y((\Delta U_1, \Delta U_2)^\top)$ in the case $\alpha = 1$ (Model C).	48
6.1	Test set pairs real and learned mappings for Model A	52
6.2	Test set pairs real and learned mappings for Model B	52
6.3	Test set pairs real and learned mappings for Model C	53
6.4	True reward functions over the 5×5 gridworld.	56
6.5	Bradley-Terry learned \mathbf{w}^{MLE} , true reward is \mathbf{R}_O	57
6.6	Bradley-Terry learned \mathbf{w}^{MLE} , true reward is \mathbf{R}_B	58
6.7	Learned \mathbf{W}_B^{MLE} , true reward is \mathbf{R}_B	59
6.8	Learned \mathbf{W}_A^{MLE} , true reward is \mathbf{R}_B	59
6.9	\mathbf{W}_A^{MLE} obtained with mini-batch gradient descent.	63
6.10	\mathbf{W}_B^{MLE} obtained with mini-batch gradient descent.	63
6.11	Learned reward matrices for increasing dataset size (Model B, true reward is \mathbf{R}_B).	64
6.12	$(1 - \epsilon)$ -correct expert (Model B)	65
6.13	$(1 - \epsilon)$ -argmax expert (Model B)	66

List of Tables

6.1	Average Jensen-Shannon divergence for all combinations of learner/expert models	54
6.2	Negative log-likelihood statistics for the Bradley-Terry model.	58
6.3	Negative log-likelihood statistics for Model B.	59
6.4	Negative log-likelihood statistics for Model A.	60
6.5	Classification metrics for Models A and B.	60
6.6	Classification metrics at increasing values of N. (average over re-runs) . . .	60
6.7	Classification metrics for $(1 - \epsilon)$ -correct expert	62
6.8	Classification metrics for $(1 - \epsilon)$ -argmax expert	62
6.9	Classification metrics for Models A and B (PyTorch implementation) . . .	63

Acknowledgements

I want to thank my family for the constant support during my studies, my friends and colleagues for their encouragement, and my advisor and co-advisors for their guidance throughout this work.

