Online Learning of Capacity-Based Preference Models

Margot Herin[1] , **Patrice Perny** [1] , **Nataliya Sokolovska**[2]
[1]Sorbonne University, CNRS, LIP6, Paris, France
[2] Sorbonne University, CNRS, LCQB, Paris, France
{margot.herin,patrice.perny}@lip6.fr, nataliya.sokolovska@sorbonne-universite.fr

Abstract

In multicriteria decision making, sophisticated decision models often involve a non-additive set function (named capacity) to define the weights of all subsets of criteria. This makes it possible to model criteria interactions, leaving room for a diversity of attitudes in criteria aggregation. Fitting a capacity-based decision model to a given Decision Maker is a challenging problem and several batch learning methods have been proposed in the literature to derive the capacity from a database of preference examples. In this paper, we introduce an online algorithm for learning a sparse representation of the capacity, designed for decision contexts where preference examples become available sequentially. Our method based on regularized dual averaging is also well fitted to decision contexts involving a large number of preference examples or a large number of criteria. Moreover, we propose a variant making it possible to include normative constraints on the capacity (e.g., monotonicity, supermodularity) while preserving scalability, based on the alternating direction method of multipliers.

1 Introduction

Multicriteria decision support tools aim to facilitate the exploration of possible trade-offs between the evaluation dimensions involved in the comparison of alternatives in a choice or ranking problem, and to recommend suitable solutions to the user [Steuer, 1986; Keeney *et al.*, 1993; Roy, 1996]. To be able to entrust the machine with the task of exploring possible solutions and making recommendations, an important step is to acquire preferential information enabling a formal model of the user's preferences to be built [Boutilier, 2013]. This step of learning preferences and the decision model is often envisaged in batch mode, i.e. it is assumed that a history of previous decisions is available, or a database of examples of pairwise comparisons, which will be exploited in its entirety to adapt a generic decision model to the user's value system [Fürnkranz and Hüllermeier, 2010; Domshlak *et al.*, 2011; Aggarwal and Fallah Tehrani, 2019].

In other contexts, particularly that of recommender systems [Zhao *et al.*, 2016], examples of preferences arrive sequentially, because they are collected progressively from recent user's feedback or answers to preference queries. In this case, for reasons of reactivity, it is generally preferable to adapt the current model to the margin using the new example (online learning), rather than restart the learning process from scratch on the set of available examples. When the entire database of examples is available but very large, it can also be efficient to consider these examples sequentially and use online learning [Shalev-Shwartz, 2012; Hoi *et al.*, 2021]. Various methods for learning decision models in batch are available in the literature on preference modeling, but the online aspect remains relatively unexploited in decision theory. The aim of this article is to propose online algorithms for learning advanced weighted aggregation functions standardly used for multicriteria decision making, including Choquet integrals and multilinear utility functions [Grabisch *et al.*, 2009].

In the setting of multicriteria aggregation, it is well known that linear models of the weighted sum type are not sufficient to cover the diversity of preferences and value systems encountered. To gain in expressiveness, we often resort to more sophisticated aggregation functions that allow a non-additive view of the importance of criteria. In this way, the weight associated with a set of criteria can be greater or less than the sum of the weights of the criteria that make it up. This makes it possible to model positive or negative synergies in the aggregation of evaluations, and thus to model interactions among criteria [Grabisch *et al.*, 2009]. The notion of weighting is then represented by a set function called *capacity*, which associates a weight to every subset [Grabisch, 2016]. The multilinear model introduced in multi-attribute utility theory [Keeney *et al.*, 1993] and Choquet's integral for multicriteria decision making [Grabisch and Labreuche, 2010] are standard examples of capacity-based multidimensional evaluation models. The task of fine-tuning the preference model is then to determine which capacity is best suited to model and explain the user's preferences.

This learning process can be complex in several respects. On the one hand, defining a capacity requires specifying a number of parameters exponential in the number of criteria

[*]The code and the proofs not included in the paper are available at https://gitlab.com/margother/OPL.

(one per subset). Moreover, these parameters are generally subject to constraints, for normative reasons. For example, in the multilinear model, as in the Choquet integral, capacity must be monotonic with respect to set-inclusion to guarantee that the associated decision model is monotonic with Pareto dominance (the improvement of a solution on one or more criteria cannot have negative repercussions in the comparison of this solution with the others). Putting such constraints on the capacity makes sure that the learned aggregation function is monotonic [Fallah Tehrani *et al.*, 2012]. Other constraints may sometimes be deemed desirable to define a suitable capacity. For example, in a context of fairness, the capacity used in the Choquet model may be required to be supermodular so as to promote solutions with balanced evaluation profiles [Lesca and Perny, 2010].

Capacity identification in decision models, possibly under monotonicity constraints, has been the subject of several studies by the past, e.g., [Grabisch *et al.*, 2008; Tehrani and Hüllermeier, 2013; Anderson *et al.*, 2014; Benabbou *et al.*, 2017]. In recent years, this issue has been very much alive in the community at the crossroads of machine learning and algorithmic decision theory., see e.g., [Beliakov and Wu, 2019; Bresson *et al.*, 2020; Pelegrina *et al.*, 2020; Kakula *et al.*, 2020a; Herin *et al.*, 2023]. However, the potential contribution of online learning to the identification of capacities remains underexplored despite a recent attempt [Kakula *et al.*, 2020b] concerning the Choquet integral without the monotonicity constraint.

In this paper, we introduce online learning algorithms suitable for a wide class of capacity-based decision models, including the Choquet integral and the multilinear model. We also propose an extension to include normative constraints on capacities such as monotonicity and supermodularity. The paper is organized as follows: Section 2 introduces the technical background needed to present the results and some illustrative examples, Section 3 proposes an online algorithm for learning a capacity without constraints, Section 4 proposes an online algorithm capable of including normative constraints (typically monotonicity or supermodularity) in the learning process. Finally, Section 5 presents numerical test results illustrating the effectiveness of the proposed approach.

2 Background and Notations

Multicriteria decision making problems are characterized by the fact that the alternatives are evaluated with respect to n dimensions representing various points of view (criteria evaluation or individual optinions) possibly conflicting each other. In the sequel, $N = \{1, \ldots, n\}$ denotes the set of criteria under consideration in the problem. The set of alternatives is denoted X. Every element $x \in X$ is represented by an evaluation vector $x = (x_1, \ldots, x_n)$ where x_i represents the value of x with respect to criterion i for $i = 1, \ldots, n$. We assume here that criterion values are all expressed on the same utility scale [0,1], 0 and 1 representing the bottom and top evaluations respectively.

Capacities and the Möbius inverse. In order to model the importance of criteria coalitions, we consider a capacity $v : 2^N \to [0,1]$, i.e., a set function such that $v(\emptyset) = 0$, $v(N) = 1$ and $v(T) \leq v(S)$ for all $S, T \subseteq N$ such that $T \subseteq S$. The latter condition, called *monotonicity with respect to set inclusion*, guarantees that no criterion has a negative contribution to the importance of the coalition to which it belongs. The explicit definition of a capacity requires 2^n parameters (one per subset of criteria) but alternative representations, sometimes more compact, can be obtained using the *Möbius inverse*. Formally, the Möbius inverse of capacity v is another set function m defined by:

$$m(S) = \sum_{T \subseteq S} (-1)^{|S \setminus T|} v(T)$$

The coefficients $m(S)$ are named Möbius masses. They completely characterize capacity v that can be recovered, for any $S \subseteq N$ by: $v(S) = \sum_{T \subseteq S} m(T)$.

By abuse of notation, v and m will also denote the vectors $(v(S)_{S \in N})$ and $(m(S)_{S \in N})$ in $\mathbb{R}^d$ with $d = 2^n$ whose components are indexed by all subsets of N ranked in lexicographic order. Due to the monotonicity constraint we have $\|m\|_0 \leq \|v\|_0$ where $\|.\|_0$ denotes the L_0 norm (i.e., the number of non-zero coefficients), thus making the Möbius representation generally more compact than the capacity itself, as pointed out in [Herin *et al.*, 2022]. An illustration of this inequality is given by k-additive capacities, i.e., capacities having null Möbius masses on all subsets including more than k elements and admitting at least a non-zero Möbius mass on a subset of size k [Grabisch, 1997]. For example, 2-additive capacities are defined using only $n + n(n-1)/2$ Möbius masses. They are used when interactions only appear on pairs of criteria but not on larger subsets.

Since $v(\{i\}) = m(\{i\})$ for every singleton, we can see that $m_{\{i,j\}} = v(\{i,j\}) - v(\{i\}) - v(\{j\})$ measures the gap to additivity in aggregating the importance of i and j. Möbius masses may be positive or negative and are used to model synergies between the elements of a coalition, e.g., in cooperative game theory and in multicriteria decision analysis [Grabisch, 2016]. When the capacity is used as a weighting function in a multicriteria decision model, the associated Möbius inverse enables to control the interactions among criteria.

Decision models based on Möbius masses. Given a capacity v and its Möbius inverse m, we consider a general aggregation function defined for any vector $x \in X$ by:

$$F(x) = \textstyle\sum_{S \subseteq N} m(S)\ \phi_S(x_S) \tag{1}$$

where x_S is the restriction of x to components belonging to S and ϕ_S is an non-linear aggregation function defining the interaction term on every subset S including more than one element ($\phi_{\{i\}}(x_i) = x_i$ for any singleton $\{i\}$). Function ϕ_S is a local aggregator measuring the quality of x restricted to subset S. Standard examples for ϕ_S are conjunctive aggregation functions like minimum or product but other functions could be considered as well. Whenever $m(S) = 0$ for some S, the interaction term ϕ_S does not appear in the model. If v is 1-additive, then all interaction terms vanish in the model and F boils down to a weighted sum: $F(x) = \sum_{i=1}^{n} m(\{i\}) x_i$. If v is 2-additive, then the weighted sum is augmented by terms modeling pairwise interactions. We obtain: $F(x) = \sum_{i=1}^{n} m(\{i\}) x_i + \sum_{i<j} m(\{i,j\}) \phi_{ij}(x_i, x_j)$.

Considering only k-additive models would be too limited. Non-null Möbius masses may appear on subsets of any size, possibly including N itself (e.g., when we want to attach a special importance to the worse component of x, using $\phi_N(x) = \min_{i \in N}\{x_i\}$). When it comes to learning model F from preference examples, we have to identify the interactions that really matter, even if it means neglecting the others to make the model more readable. This can be achieved by looking for a sparse m vector.

Function F simply defines a preference model by stating that a solution x is at least as good as y (denoted $x \succsim y$) if and only if $F(x) \geq F(y)$. Our focus on model F defined in Equation 1 is justified by several reasons. First, it covers a large class of sophisticated decision models used in multicriteria analysis. In particular, it includes the discrete Choquet integral used to model preferences based on interacting criteria [Grabisch *et al.*, 2009] and defined by:

$$C_v(x) = \sum_{i=1}^{n} \left[v(X_{(i)}) - v(X_{(i+1)})\right] x_{(i)}$$

where (.) is any permutation of N such that $x_{(i-1)} \leq x_{(i)}$, $i = 1, \ldots, n$, $X_{(i)} = \{(i), \ldots, (n)\}$ and $X_{(n+1)} = \emptyset$. Using Möbius masses, C_v can indeed be rewritten as follows [Chateauneuf and Jaffray, 1989]:

$$C_v(x) = \sum_{S \subseteq N} m(S) \min_{i \in S}\{x_i\}$$

We recognize an instance of model F where $\phi_S(x_S) = \min_{i \in S}\{x_i\}$. The Choquet integral includes various simpler decision models as special cases such as weighted averages, order statistics, ordered weighted averages [Yager, 1988] and their weighted extension [Torra, 1997]. They are therefore all instances of the F model. Similarly, the multilinear model used in multiattribute utility theory [Dyer and Sarin, 1979] and in game theory [Owen, 1972] is defined by:

$$ML_v(x) = \sum_{S \subseteq N} v(S) \prod_{i \in S} x_i \prod_{i \notin S}(1 - x_i)$$

It also reads as follows:$ML_v(x) = \sum_{S \subseteq N} m(S) \prod_{i \in S} x_i$. Here also, we recognize an instance of model F where $\phi_S(x_S) = \prod_{i \in S} x_i$.

Beyond its generality, the focus on model F is justified by its linearity in parameter m. F indeed takes the simple form of an inner product $\langle m, \phi(x) \rangle$ where $\phi(x) = (\phi_S(x), S \subseteq N)$ whose components are indexed by the subsets of N ranked in lexicographic order. Thus, $F(x)$ is a linear function of m for any given x, and any constraint of type $F(x) \geq F(y)$ associated to a preference example $x \succsim y$ for any given pair (x, y) reads $\langle m, \phi(x) - \phi(y) \rangle \geq 0$ which is a linear inequality. This also holds for indifference examples that translate to linear equations. Hence, a database of preference and indifference example translates into a linear system of equations, which significantly simplifies the learning process.

Dealing with constraints on the capacity. Monotonicity of v w.r.t. set inclusion is a necessary and sufficient condition to guarantee that Choquet and multilinear models are non-decreasing in every component and therefore that the induced preference is consistent with weak Pareto dominance (e.g., $\forall i \in N, x_i \geq y_i \Rightarrow x \succsim y$). When it comes to learning capacity from preference examples, whether in the Choquet integral, in the multilinear model, or more generally in the F model, the question arises as to how to obtain a capacity that verifies this monotonicity property. Let us first remark that preference examples may partly contribute to enforce monotonicity. For instance, in the case of Choquet and multilinear model we have $F(1_S, 0_{-S}) = v(S)$ for all $S \subseteq N$ where $(1_S, 0_{-S})$ is the vector of $\mathbb{R}^n$ whose components indexed in S equal 1, the other being equal to 0. Hence, for any pair T, S of subsets such that $T \subseteq S \subseteq N$, a preference example like $(1_S, 0_{-S}) \succsim (1_T, 0_{-T})$ is equivalent to $v(S) \geq v(T)$. Thus a capacity-based decision model that well fits such preference examples should nearly satisfy monotonicity on the pairs present in the database, However, in practice, preference are collected from past experiences and we cannot expect that all relevant (S, T) pairs are present in the preference database. Multiple violations of monotonicity are still possible. Another approach to enforce monotonicity is to explicitly include all monotonicity constraints in the learning process. The constraints can be directly expressed using Möbius masses as follows: $\sum_{T \subseteq S, T \ni i} m(T) \geq 0, \ \forall i \in S, \forall S \subseteq N$. This option will be investigated in Section 4.

Beyond monotonicity, other structural constraints on the capacity might be considered, and in particular supermodularity. A capacity v is said to be *supermodular* (or convex) if $v(S \cup T) + v(S \cap T) \geq v(S) + v(T)$ for all $S, T \subseteq N$. This condition is used in the Choquet integral to support the emergence of fair solutions in choices [Lesca and Perny, 2010]. More precisely, if the decision maker is indifferent between q solutions $x^1, \ldots, x^q$, with a supermodular v it is guaranteed that a vector obtained by convex combination of $x^1, \ldots, x^q$ will be preferred to any of the x^i's [Chateauneuf and Tallon, 2002]. Thus, softening the variations of components in vectors $x^i, i = 1, \ldots, q$ makes the decision maker better off. This condition promotes alternatives having balanced profiles.

Example 1. *If the decision maker is indifferent between (1, 0) and (0, 1), a solution like $(\frac{1}{2}, \frac{1}{2})$ will be preferred to the other two according to the Choquet model provided that the capacity is supermodular. We have indeed, $C_v(1, 0) = v(\{1\})$, $C_v(0, 1) = v(\{2\})$ and $v(\{1\}) = v(\{2\})$ since $(1, 0)$ and $(0, 1)$ are indifferent. Hence $C_v(\frac{1}{2}, \frac{1}{2}) = \frac{1}{2}v(\{1, 2\}) \geq \frac{1}{2}(v(\{1\}) + v(\{2\}))$ by supermodularity of v. Therefore $C_v(\frac{1}{2}, \frac{1}{2}) \geq v(\{1\}) = C_v(1, 0) = C_v(0, 1)$.*

Obviously, supermodularity can also be expressed using Möbius masses. The explicit consideration of supermodularity constraints in the learning process for reasons of fairness will be also addressed in Section 4.

3 Online Learning of the Capacity

3.1 Online Preference Learning

Online learning algorithms work sequentially [Shalev-Shwartz, 2012; Hoi *et al.*, 2021]. At each round t, a new instance is received and the learner makes a prediction. Then, the true label or value of the instance is received, and in the case of an incorrect prediction, the learner suffers a certain loss $l_t(m_t)$ where m_t is the current model, and $l_t : \mathcal{M} \to \mathbb{R}$ is an instantaneous loss function defined on the set of admissible models $\mathcal{M}$. One desirable property of an online algorithm is the guarantee that the cumulative loss after T rounds is close to the minimal cumulative loss one could obtain with all the instances in hand. To this end, model m_t is updated at each round t, in such a way that the regret against the best fixed model in hindsight $R_T = \sum_{t=1}^{T} l_t(m_t) - \min_{m\in\mathcal{M}} \sum_{t=1}^{T} l_t(m)$ is guaranteed to be sublinear in T, i.e., $\lim_{T\to\infty} \frac{R_T}{T} = 0$. When l_t is a convex loss function and $\mathcal{M}$ is a convex set, online convex optimization provides us with efficient algorithms that achieve sublinear regrets. For instance, Online Gradient Descent (OGD) is a simple online learning algorithm that uses the update $m_{t+1} = \Pi_{\mathcal{M}}(m_t - \eta_t g_t)$ where $g_t \in \partial l_t(m_t)$ is a subgradient of l_t evaluated at m_t, η_t is a learning rate and $\Pi_{\mathcal{M}}$ is the Euclidean projection on an admissible set $\mathcal{M}$ (i.e, $\Pi_{\mathcal{M}}(m_0) = \mathrm{argmin}_{m\in\mathcal{M}} \|m - m_0\|_2^2$). OGD is known to achieve sublinear regret R_T with a $O(\sqrt{T})$ regret bound [Zinkevich, 2003].

We now consider the online setting to learn a sparse Möbius vector m in model F from preference examples. The preference examples are supposed to be received as a stream of pairwise preference examples (x_t, y_t), where at each round t, we consider without loss of generality that $x_t \succ y_t$ (strict preference) or $x_t \sim y_t$ (indifference). To define the instantaneous loss l_t adapted to this online preference learning problem, we first consider the batch counterpart problem.

In the batch problem, we have in hand a training set of pairwise comparisons $(x_t, y_t)_{t=1}^{T}$. A natural problem formulation for preference learning is to minimize the cumulative error on the preference examples and a regularization term $\Psi(m)$ [Joachims, 2002; Tsochantaridis *et al.*, 2005; Waegeman *et al.*, 2009; Tehrani, 2021; Herin *et al.*, 2023]. This leads to the following problem:

$$\min_{m\in\mathcal{M}} \frac{1}{|\mathcal{P}|}\sum_{t\in\mathcal{P}} \epsilon_t + \frac{1}{|\mathcal{I}|}\sum_{t\in\mathcal{I}} (\epsilon_t^- + \epsilon_t^+) + \lambda\Psi(m) \qquad (2)$$
$$\begin{aligned} \langle m, \phi(x_t) - \phi(y_t)\rangle &\geq \delta - \epsilon_t,\ t\in\mathcal{P} \\ \langle m, \phi(x_t) - \phi(y_t)\rangle &\leq \delta + \epsilon_t^+,\ t\in\mathcal{I} \\ \langle m, \phi(y_t) - \phi(x_t)\rangle &\leq \delta + \epsilon_t^-,\ t\in\mathcal{I} \\ \epsilon_t, \epsilon_t^+, \epsilon_t^- &\geq 0, \qquad t = 1,\dots,T \end{aligned}$$

where $\mathcal{P}$, $\mathcal{I} \subseteq \{1,\dots,T\}$ are respectively the index set of strict preference and indifference examples, $\delta > 0$ is a discrimination threshold used to separate preference from indifference situations and $\lambda > 0$ is a hyperparameter that controls the level of regularization. Variables ϵ_t (resp. ϵ_t^+,ϵ_t^+) are error variables making flexible preference (resp. indifference) constraints.

Problem (2) amounts to minimizing the regularized loss function $\frac{1}{T}\sum_{t=1}^{T}(l_t(m) + \lambda\Psi(m))$ where loss l_t measures the violation of preference $x_t \succ y_t$ if $t\in\mathcal{P}$ or indifference $x_t \sim y_t$ if $t\in\mathcal{I}$, i.e.:

$$\begin{aligned} l_t(m) &= [\delta - \langle m, \phi(x_t) - \phi(y_t)\rangle]_+ && \text{if } t\in\mathcal{P} \\ &= [|\langle m, \phi(x_t) - \phi(y_t)\rangle| - \delta]_+ && \text{if } t\in\mathcal{I} \end{aligned} \qquad (3)$$

where for any vector $v \in \mathbb{R}^d$, $[v]_+ = (\max(0, v_i))_{i=1}^{d}$. When $\Psi(m) = \frac{1}{2}\|m\|_2^2$, Problem (2) is similar to a standard Support Vector Machine [Joachims, 2002] optimization problem. Here, to promote sparse solutions and thus obtain sparse Möbius representations of capacities, we use the well-known sparse-inducing penalty $\Psi(m) = \|m\|_1$ [Tibshirani, 1996]. Then, the batch problem of learning compact preference models naturally extends to the online setting by taking the instantaneous L_1-regularized loss $f_t(m_t) = l_t(m_t) + \lambda\|m_t\|_1$. Note that for any $t \in \{1,\dots,T\}$, f_t is a convex function.

Basic online algorithms such as OGD have been extended to handle L_1-regularized losses [Langford *et al.*, 2009; Duchi and Singer, 2009; Duchi *et al.*, 2010]. However, these gradient-descent-based algorithms show difficulties in fully exploiting the L_1-regularization and in particular provide models with high numbers of non-null coefficients [Xiao, 2009]. Nevertheless, another family of online algorithms called Regularized Dual Averaging (RDA) [Xiao, 2009] (or equivalently Follow-the-Regularized-Leader (FTLR) [Shalev-Shwartz, 2007]) is known to produce enhanced sparse models compared to gradient-descent based methods [Hoi *et al.*, 2021; Xiao, 2009]. For this reason, in the next subsection, we propose an RDA algorithm for learning compact Möbius preference representations of capacities. The benefit of using an RDA algorithm over an OGD algorithm is illustrated with numerical experiments in Section 5.

3.2 A RDA Algorithm for Capacity Learning

In this section, no constraint is put on the capacity and therefore the set of admissible Möbius vectors is assumed to be $\mathcal{M} = \mathbb{R}^d$. The case of constrained capacities is addressed in Section 4. In general, an RDA (or FTLR) algorithm consists in taking m_{t+1} as the model that minimizes the cumulative regularized loss on the past rounds [Shalev-Shwartz, 2012]. Then, contrarily to ODG that computes m_{t+1} based on m_t and the loss received at step t, RDA uses the whole history of received losses. Following the RDA principle with the regularized loss f_t to learn compact Möbius preference representations, we obtain the following model update:

$$m_{t+1} = \underset{m\in\mathcal{M}}{\mathrm{argmin}}\Big\{\frac{1}{t}\sum_{\tau=1}^{t} f_\tau(m) + \eta_t\,\|m\|_2^2\Big\} \qquad (4)$$

where $\eta_t\|m\|_2^2$ is a strongly convex regularization term and $\eta_t = \gamma/2\sqrt{t}$. With other mild assumptions, the strongly convex regularization term guarantees a $O(\sqrt{T})$ regret bound for the regularized loss function f_t [Xiao, 2009].

Unfortunately, contrarily to an OGD update, Equation (4) does not admit a closed-form solution and requires solving a quadratic program at each iteration. However, using the convexity of l_t and the subgradient definition, for any sequence of models $(m_t)_{t=1}^{T}$ and any fixed model m, we have that $\sum_{t=1}^{T}(f_t(m_t) - f_t(m)) = \sum_{t=1}^{T}(l_t(m_t) - l_t(m) + \lambda(\|m_t\|_1 - \|m\|_1)) \leq \sum_{t=1}^{T}(\langle g_t, m_t - m\rangle + \lambda(\|m_t\|_1 - \|m\|_1)) = \sum_{t=1}^{T}(\tilde{f}_t(m_t) - \tilde{f}_t(m))$. Then, the regret w.r.t. to the losses f_t is upper bounded by the regret w.r.t. the linearized loss $\tilde{f}_t$. Therefore, replacing the regularized loss function f_t by $\tilde{f}_t$ in Equation (4) also guarantees a $O(\sqrt{T})$ regret bound for f_t. Using $\tilde{f}_t$ instead of f_t, Equation (4) admits the following efficient closed-form solution [Xiao, 2009]:

$$m_{t+1} = -\frac{\sqrt{t}}{\gamma}\Big[|\bar{g}_t| - \lambda\Big]_+ \mathrm{sign}(\bar{g}_t) \qquad (5)$$

Algorithm 1

```
Parameter: $(\gamma, \lambda, T)$
$t \leftarrow 1, m_1 \leftarrow (0, \ldots, 0)$
while $t < T$ do
    receive pairwise example $(x_t, y_t)$
    compute loss gradient $g_t \in \partial l_t(m_t)$
    update average gradient $\bar{g}_t$
    $m_{t+1} \leftarrow -\frac{\sqrt{t}}{\gamma}\left[|\bar{g}_t| - \lambda\right]_+ \mathrm{sign}(\bar{g}_t)$
end while
return $m_T$
```

where for any sequence of vector $u_t \in \mathbb{R}^d$ $\bar{u}_t = \frac{1}{t}\sum_{\tau=1}^{t} u_\tau$, and for any vector $u \in \mathbb{R}^d$, $\mathrm{sign}(u)_i = 1$ if $u_i > 0$, $\mathrm{sign}(u)_i = -1$ if $u_i < 0$ and $\mathrm{sign}(u)_i = 0$ otherwise, $i = 1, \ldots, d$. Note that for $t \in \mathcal{P}$, $g_t = -(\phi(x_t) - \phi(y_t))$ if $\langle m_t, \phi(x_t) - \phi(y_t)\rangle < \delta$ and $g_t = 0$ otherwise. For $t \in \mathcal{I}$, $g_t = -(\phi(x_t) - \phi(y_t))$ if $\langle m_t, \phi(x_t) - \phi(y_t)\rangle < -\delta$, $g_t = \phi(x_t) - \phi(y_t)$ if $\langle m_t, \phi(x_t) - \phi(y_t)\rangle > \delta$ and $g_t = 0$ otherwise. The online learning algorithm based on Equation (5) for learning a compact Möbius representation of capacities is summarized in Algorithm 1.

The proposed online approach has the well-known advantage of improving the scalability of the learning task, compared with batch problem solving (2). Due to the efficient closed-form of Equation (5), Algorithm 1 applies on instances involving more than 20 criteria (millions of interactions), as shown by the results of numerical tests given in Section 5. Handling problems with such size is also possible with a recent contribution ([Herin *et al.*, 2023]) in batch mode, provided the database of preference examples is small (a few hundreds). Here, the computational complexity of Algorithm 1 is in $O(Td)$. It is still exponential in the number of criteria since $d = 2^n$ but linear in T for bounded n. This is an advantage in view of processing large-size databases.

On the other hand, Algorithm 1 does not enforce monotonicity constraints on the capacity. As suggested in Section 2, if the DM preferences are monotonic w.r.t Pareto dominance, we may observe in practice that the algorithm progressively captures the data monotonicity as new preference examples arrive (this is confirmed by our numerical tests, see Section 5). However, even though the average monotonicity violation progressively vanishes, high-amplitude and recurrent violations can occur, especially at the beginning of the online learning process. In the next section, we propose an extension of Algorithm 1 that explicitly includes monotonicity constraints and possibly other constraints such as supermodularity constraints.

4 Online Learning of Constrained Capacity

4.1 Online Learning with Constraints

Online learning with constraints usually requires a projection step to bring back the current model into the admissible set $\mathcal{M}$ at each iteration. However, projections often require heavy computational efforts and may cancel the efficiency of the unconstrained online algorithms. For this reason, projection-free online algorithms have been developed. For instance, an algorithm based on the Frank-Wolfe method [Hazan and Kale, 2012] replaces the projection step with linear programming. However, this approach is not very efficient to achieve monotonicity, due to the number of constraints required.

Other methods do not enforce the constraints at every step but use the concept of long-term constraints and guarantee a bound on the cumulative constraint violation, similarly to the regret bound [Mahdavi *et al.*, 2012; Wang and Banerjee, 2012; Jenatton *et al.*, 2016; Yu and Neely, 2020]. Among them, online ADMM methods [Wang and Banerjee, 2012; Suzuki, 2013] combine online algorithms with ADMM, a well-known iterative optimization method for batch problems that uses splitting variables to reduce the optimization problem into easier sub-problems at each iteration [Boyd *et al.*, 2011]. Online ADMM methods have been shown to produce projection-free online algorithms for linear constraints both in the context of OGD [Wang and Banerjee, 2012] and RDA [Suzuki, 2013] online algorithms. In the following, we combine ideas of both works to propose an ADMM online algorithm to learn a constrained Möbius vector in model F.

4.2 An ADMM-RDA Algorithm

Let b denote the number of constraints and $B \in \mathbb{R}^{b\times d}$ the matrix encoding the linear constraints on the capacity such as monotonicity and/or supermodularity constraints, i.e., such that the set of admissible models is $\mathcal{M} = \{m : Bm \leq 0\}$.

Example 2. *When $N = \{1, 2\}$ monotonicity and supermodularity are enforced by the system $Bm \leq 0$ with:*

$$B = \begin{pmatrix} -1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & -1 & -1 \\ -1 & 0 & -1 \\ 0 & 0 & -1 \end{pmatrix} \quad \textit{and} \quad m = \begin{pmatrix} m_1 \\ m_2 \\ m_{12} \end{pmatrix}$$

Monotonicity is guaranteed by the four first lines, and supermodularity by the fifth.

The size of matrix B is exponential in n. However, B gets sparser as n increases which allows us to resort to specialized libraries (e.g., scipy.sparse) for efficient matrix products in learning algorithms.

Let $z \in \mathbb{R}^b$ denote a vector of auxiliary splitting variables enabling the separation of loss minimization and constraint satisfaction. Then, using the linearized regularized loss $\tilde{f}_t$ as in the unconstrained case, Equation (4) can be formulated in the constrained case as follows:

$$m_{t+1} = \underset{z=Bm}{\mathrm{argmin}}\Big\{\sum_{\tau=1}^{t}\tilde{f}_\tau(m) + \eta_t\|m\|_2^2 + \mathbb{I}_B(z)\Big\} \qquad (6)$$

where $\mathbb{I}_B(z) = 0$ if $z \in \mathbb{R}^b_-$ and $\mathbb{I}_B(z) = +\infty$ otherwise. At each iteration t, the augmented Lagrangian of Problem (6) reads as follows:

$$\mathcal{L}_t(m, z, \mu) = \langle \bar{g}_t, m\rangle + \lambda\|m\|_1 + \eta_t\|m\|_2^2 + \mathbb{I}_B(z) - \langle \mu, Bm - z\rangle + \frac{\rho}{2}\|Bm - z\|_2^2$$

where $\mu \in \mathbb{R}^b$ is the vector of Lagrangian multipliers attached to the constraints, and $\rho > 0$ is a parameter that controls the

Algorithm 2

```
Parameter: $(\gamma, \lambda, \rho, T)$
 1: $t \leftarrow 1, \; m_1, \mu_1, z_1 \leftarrow (0, \ldots, 0)$
 2: while $t < T$ do
 3:     receive pairwise example $(x_t, y_t)$
 4:     compute loss gradient $g_t \in \partial l_t(m_t)$
 5:     update average gradient $\bar{g}_t$
 6:     $\bar{g}_t^{\mu} \leftarrow \bar{g}_t - B^{\mathsf{T}}(\bar{\mu}_t - \rho(B\bar{m}_t - \bar{z}_t))$
 7:     $m_{t+1} \leftarrow -\frac{\sqrt{t}}{\gamma}\left[|\bar{g}_t^{\mu}| - \lambda\right]_+ \mathrm{sign}(\bar{g}_t^{\mu})$
 8:     $z_{t+1} \leftarrow -\left[\frac{\mu_t}{\rho} - Bm_{t+1}\right]_+$
 9:     $\mu_{t+1} \leftarrow \mu_t - \rho(Bm_{t+1} - z_{t+1})$
10: end while
11: return $m_T$
```

(n, T)	(10, 500)	(15, 750)	(20, 1000)
Batch (LP)	0.92 ± 0.03	0.89 ± 0.03	–
Algo 1	0.88 ± 0.03	0.84 ± 0.03	0.79 ± 0.03

Table 1: Average accuracy over 20 simulations.

(n, T)	(10, 500)	(15, 750)	(20, 1000)
Batch (LP)	1.94 ± 0.21	246.8 ± 20.4	–
Algo 1	0.04 ± 0.01	0.7 ± 0.1	66.7 ± 1.9

Table 2: Average training times (sec.) over 20 simulations.

level of penalization. Exploiting the ADMM principle [Boyd *et al.*, 2011] for online learning, we minimize the augmented Lagrangian successively over m and z and update the Lagrangian multiplier vector μ at each iteration. This yields the following updates:

$$m_{t+1} = \underset{m}{\mathrm{argmin}}\, \mathcal{L}_t(m, \bar{z}_t, \bar{\mu}_t) \tag{7}$$

$$z_{t+1} = \underset{z}{\mathrm{argmin}}\, \mathcal{L}_t(m_{t+1}, z, \mu_t) \tag{8}$$

$$\mu_{t+1} = \mu_t - \rho(Bm_{t+1} - z_{t+1}) \tag{9}$$

Note that in Equation (7), z and λ are set to their average values over the past rounds, allowing to keep memory of the past constraint violations [Suzuki, 2013]. Also, due to the term $\frac{\rho}{2}\|Bm - z\|_2^2$ in the augmented Lagrangian $\mathcal{L}_t$, the first equation (7) does not admit an efficient closed-form solution. Constraint matrix B indeed induces non-separability of the optimization problem w.r.t the components of m and thus bans us from getting a closed-form solution similar to Equation (5) in Algorithm 1. To bypass this issue, we use a linearization that is standard for ADMM methods [Deng and Yin, 2016; Wang and Banerjee, 2012; Suzuki, 2013] and that consists in using the linearized augmented Lagrangian $\tilde{\mathcal{L}}_t = \mathcal{L}_t - \frac{\rho}{2}\|B(m - \bar{m}_t)\|_2^2$ in Equations (7),(8), and (9). This requires taking γ sufficiently large so that $\eta_t\|m\|_2^2 - \frac{\rho}{2}\|B(m - \bar{m}_t)\|_2^2$ is still a strongly convex regularization. Using this linearization, we obtain:

Proposition 1. *Equations (7-9) where $\mathcal{L}_t$ is replaced by $\tilde{\mathcal{L}}_t$ admit closed-form solutions given in Algorithm 2 line 7-9.*

The proof is given in the supplementary material. The resulting online learning process is given in Algorithm 2 and its benefit for retrieving monotonic capacity is illustrated in the next section with numerical experiments.

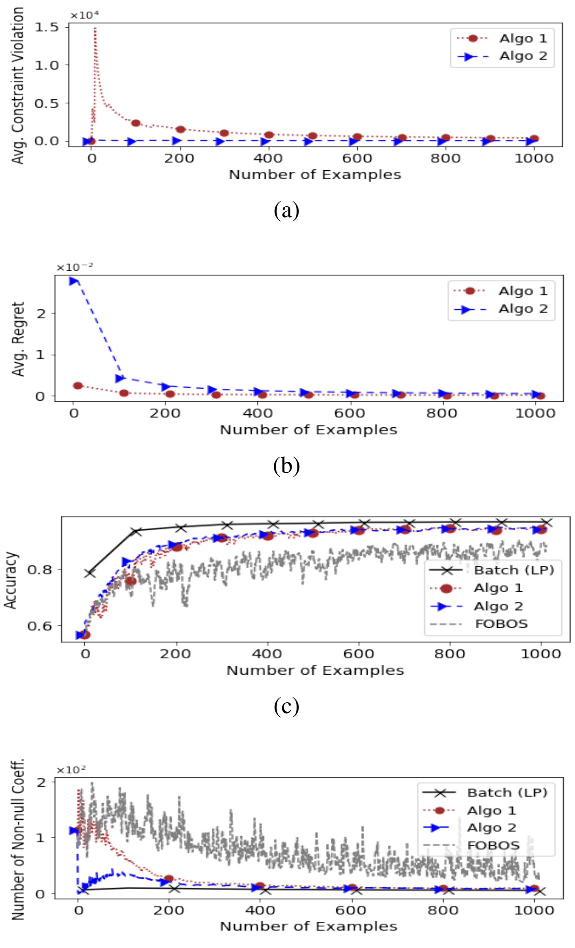
5 Numerical Tests

In this section, we conduct numerical tests using synthetic preference data We generate preference data by randomly drawing sparse (with few non-null coefficients) normalized Möbius vector m associated with monotonic capacities and pairs of alternatives $x_t, y_t \in [0, 1]^n$. Then, after comparison of the perturbed overall values $\langle m, \phi(x_t)\rangle + \epsilon_x$ and $\langle m, \phi(y_t)\rangle + \epsilon_y$ (where ϵ_x is a centered Gaussian noise with standard error $\sigma = 0.03$), we obtain preference or indifference examples. In all the experiments, we test our algorithms on the learning of Choquet Integral, and thus we generate data using $\phi(x_t) = (\min_{i \in S}\{x_i\})_{S \subseteq N}$ but the tests could be presented with the ML model with similar results.

In the first experiment, we show the practical efficiency of Algorithm 1 compared to batch problem (2) solved with linear programming (denoted Batch(LP)). The L_1-regularization parameter λ is set to 0.01 for both methods and for Algorithm 1, γ is set to 10^3. In Table 1 and 2 we compare the average accuracy and training times over 20 simulations of both methods for a growing number of criteria n. The accuracy is computed as the average proportion of correctly predicted preferences within a test set containing 500 preference examples. The number of preference examples T increases linearly with n. We observe that for 10 and 15 criteria, Algorithm 1 reaches accuracy values close to the one obtained with the batch solution (at most 5% lower) while having significantly lower training times. Finally, for 20 criteria (millions of possible criteria interactions), it provides a solution in around 1 minute that approximately reaches 80% of accuracy while no solution can be obtained in batch using linear programming.

In the second experiment, we first compare Algorithm 1 and Algorithm 2 in the retrieval of monotonic capacities. The number of criteria is set to $n = 10$ and the total number of preference examples is set to $T = 1000$. Preference examples are generated as in the previous experiment; hyperparameters λ and γ are unchanged and $\rho = 1$ for Algorithm 2. Figure 1 (a) represents the average monotonicity violation computed as $\frac{1}{t}\sum_{\tau=1}^{t}\|[Bm_\tau]_+\|_2^2$ where B is the matrix encoding the monotonicity constraints. We observe that Algorithm 1 highly violates monotonicity constraints before $t = 200$ examples while we obtain a nearly null average violation for Algorithm 2 at any t. Remark that Algorithm 1 progressively captures monotonicity as it receives preference examples. In Figure 1 (b), we show the average regret $\frac{1}{t}R_t = \frac{1}{t}\sum_{\tau=1}^{t} f_\tau(m_\tau) - \min_{m \in \mathcal{M}} \frac{1}{t}\sum_{\tau=1}^{t} f_\tau(m)$ w.r.t. the number of preference examples t. The optimal value $\min_{m \in \mathcal{M}} \sum_{\tau=1}^{t} f_\tau(m)$ is computed with linear programming. We observe that both algorithms provide sequences of learned models m_t with vanishing average regrets.

Then, we show the performances of both Algorithms 1



(a)

(b)

(c)

(d)

Figure 1: Avg. constraint violation (a), Avg. regret (b), , accuracy (c), and number of non-null coefficients (d) w.r.t. the number of preference examples t.

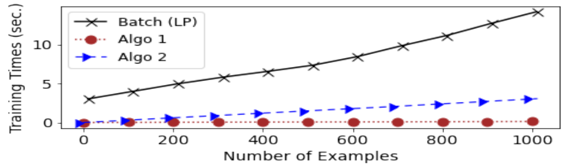
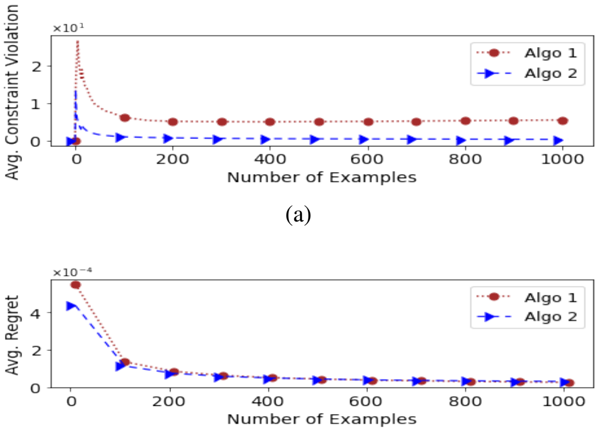


Figure 2: Training times (sec.) w.r.t. the number of preference ex. t.

and 2 in terms of accuracy and number of non-null coefficients respectively in Figure 1 (c) and (d). We compare their performances with the Batch(LP) method and with the FOBOS algorithm (an OGD-type algorithm for handling L_1-regularized loss) implemented using the loss l_t and a learning rate $\eta_t = \eta_1/\sqrt{t}$ with η_1 set at the recommended value in [Duchi and Singer, 2009]. We observe that FOBOS suffers from instability and produces less compact models. In contrast, Algorithms 1 and 2 quickly reduce the number of non-null coefficients to some dozens. Concerning accuracy, we observe that Algorithm 2 achieves the same performance as Algorithm 1 while providing a better control of motonicity. The accuracy of both Algorithms 1 and 2 are slightly below the one obtained with Batch(LP). However, the associated training time curves presented in Figure 2 reveal the efficiency of the online algorithms compared to batch (LP). In particular Algorithm 1 achieves these results in a near null training time. Algorithm 2 achieves intermediate times between Algorithm 1 and Batch(LP).

In the third experiment, we assess the benefit of using Algorithm 2 to learn both monotonic and supermodular capacities. More precisely, we compare the average violation of constraints for both Algorithms 1 and 2 in Figure 3 (a) and the average regret in Figure 3 (b). The advantage of Algorithm 2 in terms of constraint respect is also clear when supermodularity is required in addition to monotonicity.



(a)

(b)

Figure 3: Avg. constraint violation (a), avg. regret (b) w.r.t. the number of preference ex. t.

6 Conclusion

We have proposed online algorithms to efficiently learn the capacity in a large class of non-linear aggregation functions (but linear in the capacity), including the well-known Choquet and multilinear models. These algorithms not only allow a decision model to be adapted to a stream of preference examples, but can also be used in place of batch learning methods, with an advantage in terms of scalability confirmed by our tests. We have also addressed the inclusion of normative constraints restricting the set of admissible capacities in the online learning process.

An interesting follow-up to this work would be to investigate the potential benefit of active selection of the next example in this online process. Further contributions could involve finding equivalents of the proposed approach for models beyond the class represented by the F model. Other aggregation functions based on different algebraic operations can indeed be used to combine capacities and values. For example, Sugeno's integral uses $(\max, \min)$ operations instead of $(+, \times)$ [Sugeno, 1977]. The main challenge in going beyond F will then be to overcome the loss of linearity of the model with respect to the capacity and its Möbius inverse m.

Acknowledgments

We gratefully acknowledge the financial support provided by SCAI (Sorbonne Center for AI) for this research.

References

[Aggarwal and Fallah Tehrani, 2019] Manish Aggarwal and Ali Fallah Tehrani. Modelling human decision behaviour with preference learning. *INFORMS Journal on Computing*, 31(2):318–334, 2019.

[Anderson *et al.*, 2014] Derek T. Anderson, Stanton R. Price, and Timothy C. Havens. Regularization-based learning of the Choquet integral. In *FUZZ-IEEE*, pages 2519 – 2526, 2014.

[Beliakov and Wu, 2019] Gleb Beliakov and Jian-Zhang Wu. Learning fuzzy measures from data: simplifications and optimisation strategies. *Information Sciences*, 494:100–113, 2019.

[Benabbou *et al.*, 2017] Nawal Benabbou, Patrice Perny, and Paolo Viappiani. Incremental elicitation of Choquet capacities for multicriteria choice, ranking and sorting problems. *Artificial Intelligence*, 246:152–180, 2017.

[Boutilier, 2013] Craig Boutilier. Computational decision support: Regret-based models for optimization and preference elicitation, 2013.

[Boyd *et al.*, 2011] Stephen Boyd, Neal Parikh, Eric Chu, Borja Peleato, Jonathan Eckstein, et al. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends® in Machine learning*, 3(1):1–122, 2011.

[Bresson *et al.*, 2020] Roman Bresson, Johanne Cohen, Eyke Hüllermeier, Christophe Labreuche, and Michèle Sebag. Neural representation and learning of hierarchical 2-additive Choquet integrals. In *IJCAI*, pages 1984–1991, 2020.

[Chateauneuf and Jaffray, 1989] Alain Chateauneuf and Jean-Yves Jaffray. Some characterizations of lower probabilities and other monotone capacities through the use of möbius inversion. *Mathematical social sciences*, 17(3):263–283, 1989.

[Chateauneuf and Tallon, 2002] Alain Chateauneuf and Jean-Marc Tallon. Diversification, convex preferences and non-empty core in the Choquet expected utility model. *Economic Theory*, 19:509–523, 2002.

[Deng and Yin, 2016] Wei Deng and Wotao Yin. On the global and linear convergence of the generalized alternating direction method of multipliers. *Journal of Scientific Computing*, 66:889–916, 2016.

[Domshlak *et al.*, 2011] Carmel Domshlak, Eyke Hüllermeier, Souhila Kaci, and Henri Prade. Preferences in ai: An overview. *Artificial Intelligence*, 175(7-8):1037–1052, 2011.

[Duchi and Singer, 2009] John Duchi and Yoram Singer. Efficient online and batch learning using forward backward splitting. *The Journal of Machine Learning Research*, 10:2899–2934, 2009.

[Duchi *et al.*, 2010] John C Duchi, Shai Shalev-Shwartz, Yoram Singer, and Ambuj Tewari. Composite objective mirror descent. In *COLT*, volume 10, pages 14–26. Citeseer, 2010.

[Dyer and Sarin, 1979] James S Dyer and Rakesh K Sarin. Measurable multiattribute value functions. *Operations research*, 27(4):810–822, 1979.

[Fallah Tehrani *et al.*, 2012] Ali Fallah Tehrani, Weiwei Cheng, Krzysztof Dembczyński, and Eyke Hüllermeier. Learning monotone nonlinear models using the Choquet integral. *Machine Learning*, 89:183–211, 2012.

[Fürnkranz and Hüllermeier, 2010] Johannes Fürnkranz and Eyke Hüllermeier. Preference learning and ranking by pairwise comparison. In *Preference learning*, pages 65–82. Springer, 2010.

[Grabisch and Labreuche, 2010] Michel Grabisch and Christophe Labreuche. A decade of application of the Choquet and Sugeno integrals in multi-criteria decision aid. *Annals of Operations Research*, 175(1):247–286, 2010.

[Grabisch *et al.*, 2008] Michel Grabisch, Ivan Kojadinovic, and Patrick Meyer. A review of methods for capacity identification in Choquet integral based multi-attribute utility theory: Applications of the Kappalab R package. *European journal of operational research*, 186(2):766–785, 2008.

[Grabisch *et al.*, 2009] Michel Grabisch, Jean-Luc Marichal, Radko Mesiar, and Endre Pap. *Aggregation functions*, volume 127. Cambridge University Press, 2009.

[Grabisch, 1997] Michel Grabisch. K-order additive discrete fuzzy measures and their representation. *Fuzzy sets and systems*, 92(2):167–189, 1997.

[Grabisch, 2016] Michel Grabisch. *Set functions, games and capacities in decision making*, volume 46. Springer, 2016.

[Hazan and Kale, 2012] Elad Hazan and Satyen Kale. Projection-free online learning. In *Proceedings of the 29th International Coference on International Conference on Machine Learning*, pages 1843–1850, 2012.

[Herin *et al.*, 2022] Margot Herin, Patrice Perny, and Nataliya Sokolovska. Learning sparse representations of preferences within Choquet expected utility theory. In *Uncertainty in Artificial Intelligence*, pages 800–810, 2022.

[Herin *et al.*, 2023] Margot Herin, Patrice Perny, and Nataliya Sokolovska. Learning preference models with sparse interactions of criteria. In *Proc. of IJCAI*, pages 3786–3794, 2023.

[Hoi *et al.*, 2021] Steven CH Hoi, Doyen Sahoo, Jing Lu, and Peilin Zhao. Online learning: A comprehensive survey. *Neurocomputing*, 459:249–289, 2021.

[Jenatton *et al.*, 2016] Rodolphe Jenatton, Jim Huang, and Cédric Archambeau. Adaptive algorithms for online convex optimization with long-term constraints. In *International Conference on Machine Learning*, pages 402–411. PMLR, 2016.

[Joachims, 2002] Thorsten Joachims. Optimizing search engines using clickthrough data. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 133–142, 2002.

[Kakula *et al.*, 2020a] Siva K Kakula, Anthony J Pinar, Timothy C Havens, and Derek T Anderson. Choquet integral ridge regression. In *2020 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*, pages 1–8. IEEE, 2020.

[Kakula *et al.*, 2020b] Siva K Kakula, Anthony J Pinar, Timothy C Havens, and Derek T Anderson. Online learning of the fuzzy Choquet integral. In *2020 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, pages 608–614. IEEE, 2020.

[Keeney *et al.*, 1993] Ralph L Keeney, Howard Raiffa, and Richard F Meyer. *Decisions with multiple objectives: preferences and value trade-offs*. Cambridge university press, 1993.

[Langford *et al.*, 2009] John Langford, Lihong Li, and Tong Zhang. Sparse online learning via truncated gradient. *Journal of Machine Learning Research*, 10(3), 2009.

[Lesca and Perny, 2010] Julien Lesca and Patrice Perny. LP solvable models for multiagent fair allocation problems. In *ECAI 2010 Proceedings*, pages 393–398, 2010.

[Mahdavi *et al.*, 2012] Mehrdad Mahdavi, Rong Jin, and Tianbao Yang. Trading regret for efficiency: online convex optimization with long term constraints. *The Journal of Machine Learning Research*, 13(1):2503–2528, 2012.

[Owen, 1972] Guillermo Owen. Multilinear extensions of games. *Management Science*, 18(5-part-2):64–79, 1972.

[Pelegrina *et al.*, 2020] Guilherme Dean Pelegrina, Leonardo Tomazeli Duarte, Michel Grabisch, and João Marcos Travassos Romano. The multilinear model in multicriteria decision making: The case of 2-additive capacities and contributions to parameter identification. *European Journal of Operational Research*, 282(3):945–956, 2020.

[Roy, 1996] Bernard Roy. *Multicriteria methodology for decision aiding*, volume 12. Springer Science & Business Media, 1996.

[Shalev-Shwartz, 2007] Shai Shalev-Shwartz. *Online learning: Theory, algorithms, and applications*. PhD thesis, Hebrew University, 2007.

[Shalev-Shwartz, 2012] Shai Shalev-Shwartz. Online learning and online convex optimization. *Foundations and Trends® in Machine Learning*, 4(2):107–194, 2012.

[Steuer, 1986] Ralph. E. Steuer. *Multiple Criteria Optimization: Theory, Computation and Application*. John Wiley, New York, 546 pp, 1986.

[Sugeno, 1977] Michio Sugeno. *Fuzzy measures and fuzzy integrals: A survey*, page 89–102. North-Holland, Amsterdam, 1977.

[Suzuki, 2013] Taiji Suzuki. Dual averaging and proximal gradient descent for online alternating direction multiplier method. In *International Conference on Machine Learning*, pages 392–400. PMLR, 2013.

[Tehrani and Hüllermeier, 2013] Ali Fallah Tehrani and Eyke Hüllermeier. Ordinal Choquistic regression. In *EUSFLAT*, 2013.

[Tehrani, 2021] Ali Fallah Tehrani. The Choquet kernel on the use of regression problem. *Information Sciences*, 556:256–272, 2021.

[Tibshirani, 1996] Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (methodological)*, 58(1):267 – 88, 1996.

[Torra, 1997] Vicenç Torra. The weighted owa operator. *International Journal of Intelligent Systems*, 12(2):153–166, 1997.

[Tsochantaridis *et al.*, 2005] Ioannis Tsochantaridis, Thorsten Joachims, Thomas Hofmann, Yasemin Altun, and Yoram Singer. Large margin methods for structured and interdependent output variables. *Journal of machine learning research*, 6(9), 2005.

[Waegeman *et al.*, 2009] Willem Waegeman, Bernard De Baets, and Luc Boullart. Kernel-based learning methods for preference aggregation. *4OR*, 7(2):169–189, 2009.

[Wang and Banerjee, 2012] Huahua Wang and Arindam Banerjee. Online alternating direction method. In *Proceedings of the 29th International Coference on International Conference on Machine Learning*, pages 1699–1706, 2012.

[Xiao, 2009] Lin Xiao. Dual averaging method for regularized stochastic learning and online optimization. *Advances in Neural Information Processing Systems*, 22, 2009.

[Yager, 1988] Ronald R Yager. On ordered weighted averaging aggregation operators in multicriteria decisionmaking. *IEEE Transactions on systems, Man, and Cybernetics*, 18(1):183–190, 1988.

[Yu and Neely, 2020] Hao Yu and Michael J Neely. A low complexity algorithm with o (t) regret and o (1) constraint violations for online convex optimization with long term constraints. *Journal of Machine Learning Research*, 21(1):1–24, 2020.

[Zhao *et al.*, 2016] Zhou Zhao, Hanqing Lu, Deng Cai, Xiaofei He, and Yueting Zhuang. User preference learning for online social recommendation. *IEEE Transactions on Knowledge and Data Engineering*, 28(9):2522–2534, 2016.

[Zinkevich, 2003] Martin Zinkevich. Online convex programming and generalized infinitesimal gradient ascent. In *Proceedings of the 20th international conference on machine learning (icml-03)*, pages 928–936, 2003.