PADERBORN UNIVERSITY
The University for the Information Society

DA2PL'2016

From Multiple Criteria Decision Aid to Preference Learning

Proceedings of the DA2PL'2016 EURO Mini Conference

Róbert Busa-Fekete Eyke Hüllermeier Vincent Mousseau Karlson Pfannschmidt

EURO
The Association of European Operational Research Societies

SFB 901
ON - THE - FLY COMPUTING

GDRI ALGODEC

Preface

This volume comprises the papers presented at DA2PL'2016, the EURO Mini Conference "From Multicriteria Decision Aid to Preference Learning" held in Paderborn, Germany, on November 7–8, 2016.

Following the two previous editions of this event (DA2PL'2014 and DA2PL'2012), the goal of the conference was to bring together researchers from decision analysis and machine learning. DA2PL aims at providing a forum for discussing recent advances and identifying new research challenges in the intersection of both fields, thereby supporting a cross-fertilisation of these disciplines.

The notion of "preferences" has a long tradition in economics and operational research, where it has been formalised in various ways and studied extensively from different points of view. Nowadays, it is a topic of key importance in fields such as game theory, social choice, and the decision sciences, including decision analysis and multicriteria decision aiding. In these fields, much emphasis is put in properly modelling a decision maker's preferences, and on deriving and (axiomatically) characterizing rational decision rules.

In machine learning, like in artificial intelligence and computer science in general, the interest in the topic of preferences arose much more recently. The emerging field of preference learning is concerned with methods for learning preference models from explicit or implicit preference information, which are typically used for predicting the preferences of an individual or a group of individuals in new decision contexts. While research on preference learning has been specifically triggered by applications such as "learning to rank" for information retrieval (e.g., Internet search engines) and recommender systems, the methods developed in this field are useful in many other domains as well.

Obviously, preference modelling and decision analysis on the one side and preference learning on the other side can ideally complement and mutually benefit from each other. In particular, the suitable specification of an underlying model class is a key prerequisite for successful machine learning, that is to say, successful learning presumes appropriate modelling. Likewise, data-driven approaches for preference modelling and preference elicitation are becoming more and more important in decision analysis nowadays, mainly due to large scale applications, the proliferation of semi-automated computerised interfaces and the increasing availability of preference data.

We thank the Program Committee and the additional reviewers for their timely and thorough participation in the review process. Moreover, we express our gratitude to our sponsors: the Association of European Operational Research Societies (EURO), the Collaborative Research Centre "On-the-Fly Computing" at Paderborn University, and the GDRI ALGODEC. They allowed us to set the conference fee to an affordable price and provide special support to PhD students. Last but not least, we acknowledge the support of EasyChair in the submission, review, and proceedings creation processes.

Paderborn/Paris
November 2016

Róbert Busa-Fekete
Eyke Hüllermeier
Vincent Mousseau
Karlson Pfannschmidt

Organizing Committee

General Chair	Eyke Hüllermeier
Program Chairs	Róbert Busa-Fekete
	Vincent Mousseau
Local Chair	Karlson Pfannschmidt

Program Committee

Jamal Atif	Raymond Bisdorff
Weiwei Cheng	Yann Chevaleyre
Stéphan Clémençon	Yves De Smet
Krzysztof Dembczynski	Sébastien Destercke
Luis Dias	Raphaël Féraud
Joachim Giesen	Michel Grabisch
Salvatore Greco	Alain Guénoche
Wojciech Kotlowski	Christophe Labreuche
Jérôme Lang	Eneldo Loza Mencía
Thibaut Lust	Lucas Maestro
Thierry Marchant	Brice Mayag
Jérôme Mengin	Patrick Meyer
Wassila Ouerdane	Meltem Ozturk
Patrice Perny	Marc Pirlot
Antoine Rolland	Ammar Shaker
Eric Sibony	Balazs Szorenyi
Bruno Teheux	Mikhail Timonin
Evgeni Tsivtsivadze	Alexis Tsoukias
Tanguy Urvoy	Aida Valls
Paolo Viappiani	Peter Vojtas
Willem Waegeman	Paul Weng
Masrour Zoghi	

List of Papers

Simulating new multicriteria data from a given data set

Jairo Cugliari[1] and **Antoine Rolland**[2]

Abstract. Several methods have been proposed in the past decades to deal with Multicriteria Decision Aiding (MCDA) problems. Even if the axiomatic foundations of these methods are generally well-known, comparing the different methods or simply analysing the results produced by the methods on real-life problems is arduous as there is a lack of benchmark MCDA datasets in the literature. A solution is therefore to simulate new MCDA dataset examples. But the analysis of real-world examples show that one must deal with data that are lightly linked, and then it is important to take into account dependency between variables when simulating new datasets. We propose in this paper three different approaches to simulate new data based on existing small datasets. We describe these methods, we propose a quality analysis of the results, and we experiment the methods on different examples from the literature.

1 Introduction

There is in Multicriteria Decision Aiding (MCDA) a great number of very specific methods to be proposed, with multiple variants. Testing these methods on several situations, using real datasets, should improve our understanding of advantages and inconveniences of each method, even if an axiomatic analysis has been done. But sometimes in real-world cases, only very few data are available; for example, from an preference learning point of view, the dataset should be so limited that it is too small to be divided into a test subset and a validation subset. Researchers should also desire to have more data to test the proposed methods. Therefore, as already pointed out in [4], there is a need of simulated multicriteria datasets to be used by the MCDA community, either for benchmarking or just to improve the understanding of each method. This paper deals with the simulated datasets issue for MCDA. Our goal is to propose one or several method to simulate new multicriteria data from an existing dataset. Simulated data should be as similar as possible to the initial dataset. Therefore, the proposed methods are supposed to be able to capture the specific structure of the initial dataset (if any), and then generate new data on demand.

Good practices in MCDA point out the fact, among others, that values on criteria should be as statistically independent as possible. Please note that this statistical independence is not the preference independence between criteria (see [7]). It just means that the values taken on different criteria should not be correlated (linearly or not) to avoid redundancy. Each time we mention independence in this paper will refer to statistical indepenence. ut in real life the values taken by an alternative on different criteria are generally not totally independent. Therefore, multicriteria data cannot be well simulated using only statistical independent sampling on each variable. The problem is then to model the correlation between criteria in a plausible way.

[1] Université de Lyon, ERIC Labs, email: Jairo.Cugliari@univ-lyon2.fr
[2] Université de Lyon, ERIC Labs, email: Antoine.Rolland@univ-lyon2.fr

In [4] we introduced a statistical approach to overcome this difficulty using copulas. In the present paper, we first propose in section 2 a new simulation method using Principal Component Analysis (PCA), and we present also the simulation method based on copulas. In section 3, we propose a quality analysis of the simulation results regarding the three methods (copulas, PCA and independent sampling). Last we experiment these methods on real datasets and show that PCA and copulas-based simulators lead to simulated datasets of higher quality than simple independent samples.

2 Generating methods

We describe in this section the three frameworks we use to create simulated MCDA data from available dataset that we describe by a matrix $\mathbf{X}$ with n rows and p columns. Each column represent an criterion (variable). Each row represent an alternative. The first method does not take into account any dependence structure of $\mathbf{X}$ and so simulate values on each criterion independently from others. While this approach can be useful in some case, in real life applications the columns of $\mathbf{X}$ can not be considered totally independent. Therefore, we incorporate this element on the simulation scheme for the second an third method.

2.1 Simulation by independent draws

The first framework uses the inverse of the classical probability integral transform. That is, if a random variable say Y has a distribution function F_Y, then one can simulate realizations of Y in two steps. First, one draws a realization u from a standard uniform law (i.e. with support on the unit interval). Then, the realization y for Y are obtained by doing $y = F_Y^{-1}(u)$, where

$$F_y^{-1}(u) = \inf\{y : F_Y(u) \geq y\} \tag{1}$$

is a generalized inverse of F (since the distribution functions are weakly monotonic and right-continuous).

Our starting point is the set $y_1, \ldots, y_n$ of size n containing realisations of the target variable with an unknown distribution function. Therefore, we estimate F_y by means of the empirical distribution function,

$$\hat{F}_Y(y) = \frac{1}{n} \sum_{i=1}^{n} \mathbf{1}_{\{y_i \leq y\}}, \tag{2}$$

which is, for each y a simple average of indicators. Graphically, the empirical distribution function is a stepwise function with jumps at the observed realizations $y_1, \ldots, y_n$ an so inducing a restriction to the realization : only observed data points can be obtained with the simulation. In order to relax this constraint we apply a additional smoothing step to obtain from (2) a continuous version using interpolation splines. Then, one need to simulate standard uniform realizations u and apply (1) replacing F_Y by $\hat{F}_Y$.

2.2 Independent draws on latent factors

The independent draws approach is somewhat disappointing when applied to MCDA framework, because the dependence between the alternatives is not considered. In order to overcome with this drawback one may consider a simple transformation of the dataset $\mathbf{X}$ to create latent factors over which the independent hypothesis will be more reasonable. Actually, if the dependence structure of $\mathbf{X}$ is only linear, one may rely on the principal components analysis to extract orthogonal variables which explain the best the variability on the data. Moreover, if $\mathbf{X}$ followed a gaussian distribution this approach would generate independent factors. Our aim is not to push so far the hypothesis but to have a simple approach that will overcome the problems mentioned before.

Concretely, we decompose X using the classical principal component analysis to get a new matrix $\mathbf{F}$. We keep the the barycentre and scale of the original dataset by an appropriate centring and standardization on the columns of $\mathbf{X}$.

Then, we apply the independent draw approach on the columns of $\mathbf{F}$ in order to get a simulated set of latent factors. Using the well known reconstruction formula for PCA, the latent factors are used to get a simulated centred and standardized version of X. Finally, the original barycentre and scales are incorporated to yield on the simulated version of $\mathbf{X}$.

Notice that while many matrix decomposition schemes exists, most of them can not be used because they lack off a reconstruction formula (i.e. Independent Components Analysis).

2.3 Simulation through copula

Simulation through copula on the MCDA framework has been recently proposed by [4]. Following the reference, we describe briefly the procedure without given the technical details.

In a nutshell a copula is a multivariate cumulative distribution function which has all its margins uniformly distributed on the unit interval (see [6] for a more formal presentation of the subject). Intuitively, a probabilistic multivariate structure can then be viewed as the coupling of the marginal behaviour by means of a copula C.

The pair-copula construction has been proposed to avoid some problems that arise on high dimension datasets (large p). Then the multivariate joint density of X is decomposed into a cascade of building blocks called pair-copula. Let f be the joint density of X which is factorized (uniquely up to a relabelling of the elements of X) as

$$f(y_1, \ldots, y_n) = f(y_p) f(y_{p-1}|y_n) \ldots f(y_1|y_2, \ldots, y_p). \quad (3)$$

Then, one can write each of the conditional densities on (3) using the copula recursively which yields on this general expression for a generic element y_i of X given a generic conditioning vector v

$$\begin{aligned} f(y_i|v) =& c_{y_i, v_j|v_{-j}}\left(F(y_i|v_{-j}), F(v_j|v_{-j})\right) \\ & \times f(y_i|v_{-j}). \end{aligned} \quad (4)$$

In last expression we use the notation v_j for the j-th element of v and v_{-j} for all the elements of v but v_j.

Vines copulas have been proposed to classify alternatives factorizations into a structured graphical model. This construction allows highly flexible decompositions of the (possibly high) dimensional distribution of X because each pair-copula can be chosen independently from the others. The iterative decomposition provided by the pair-copula construction is then arranged into a set of linked trees (acyclic connected graph). Two special schemes are usually used: C-vines (canonical vines) and D-vines. In the former one, a dependent variable is identified and chosen to be the root of the tree. In the following tree, the dependence will be computed conditional on this first variable and so on. In the latter scheme, a variable ordering is chosen. Then on the first tree one models the dependence of each of the consecutive pairs of variables. The following tree will model the dependence of the remaining pairs, conditional on the those that were already modelled.

Simulation of copula data (i.e. n-variate data with uniformly distributed marginals) can be done using the probability integral transform. It is convenient to define the h-function

$$h(y|v, \theta) = \frac{\partial^d C_{y, v_j|v_{-j}}(F(y|v_j), F(y|v_{-j}), |\theta)}{\partial F(v_j|v_{-j})}, \quad (5)$$

where θ is a parameter vector associated to the decomposition level. The h-function is the conditional distribution of x given v and we let $h^{-1}(u|v, \theta)$ be its inverse with respect to u, i.e. the inverse of the cumulative conditional distribution. The simulation for the vine is as follows. First sample n uniformly distributed random variables $w_1, w_2, \ldots, w_n$. Then use the probability integral transform of the corresponding conditional distribution:

$$\begin{aligned} y_1 &= w_1, \\ y_2 &= F^{-1}(w_2|y_1), \\ y_3 &= F^{-1}(w_3|y_1, y_2), \\ &\ldots \\ y_p &= F^{-1}(w_{n-1}|y_1, \ldots, y_{p-1}). \end{aligned}$$

At each step, the computation of the inverse conditional distribution is made through the (inverse) h-function.

3 Experiments

3.1 Experiment process

We present in this section the testing process of our experiments. Our goal is to compare the results of simulating data using the three different methods previously stated. For this purpose, we first need to state a quality index of the simulation data.

3.1.1 Quality index

We state that data are correctly simulated if it is not possible to distinguish the real data and the simulated ones. So we need a tool that is able to distinguish two different distributions, such as a statistical multivariate goodness-of-fit test. But as pointed out by McAssey [5], non-parametric goodness-of-fit test that can be used in practice is something very hard to find. We then choose to use the goodness-of-fit test proposed by Szkely and Rizzo [8], based on a geometric approach, and implemented in R. The null hypothesis H_0 is "the two multivariate distributions are the same", versus hypothesis H_1 "the two multivariate distributions are the different". The test returns a p-value : if this p-value is less than a fixed threshold, then the difference between the two distribution is said to be statistically significant, and then the simulated data cannot be considered to have the same distribution as the real ones. The p-value can be seen as a quality index : the greater the p-value, the greater the simulation quality. Of course the p-value will change for each simulated data set. Therefore, the following testing process has been established to compare the different simulation methods:

1. for each real data set, for each simulation method, simulate n new data sets;
2. for each simulated data set, compute the p-value of the goodness-of-fit test comparing real and simulated data sets;
3. plot the boxplot of all the p-values.

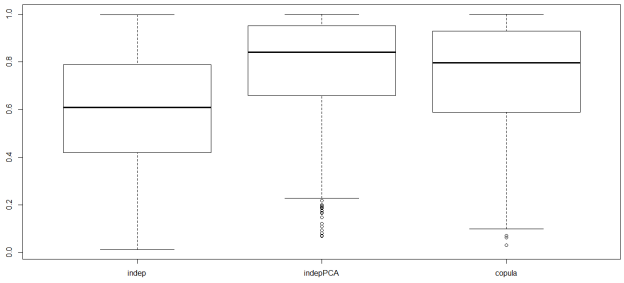


Figure 1: Example of a quality boxplot

On Figure 1, one can observe that real and simulated data can easily be confused, as more than 75% of the simulated data sets have a p-value greater than 0.5 for the confusion test (remember that generally the threshold to reject H_0, equal distribution hypothesis, is a p-value less than 0.05). Other comparison processes have been studied. Especially, we tried to use supervised and unsupervised classification methods to determine whether real and simulated data can be distinguished using machine learning. However, these methods (SVM, k-means, random forest) did not managed to separate real and simulated data when the data have the same margin distribution, which is always the case here by construction.

3.1.2 Comparing different generating processes

In the following, we will use boxplots like the one presented in Figure 1 to compare different simulation processes. If the distribution of p-values obtained by method A is higher than the distribution of p-values obtained by method B, it means that method A gives better simulations than method B, as it should be more difficult to reject the equal distribution hypothesis in case A than in case B. We will compare the three different methods proposed in section 2: "indep" corresponds to independent variables samples, "PCA" corresponds to independent variables samples on the different PCA axes, and "copula" uses copula to learn the links between variables.

3.1.3 Influence factors

Different factors can have an influence on the quality of the data simulation:

1. the number of variables
2. the size of the learning set
3. the strength of the link between variables

The effect of each of these factors will be tested independently. The testing process is the following:

1. for each value of the tested variable, generate 30 different datasets using a multivariate normal distribution with a specified correlation coefficient between the variable (through the Cholewsky matrix). Then exponential (for one variable) and log (for another variable) transformations are used to produce a dataset with a non-linear controlled link between variables. These are the reference datasets.
2. for each reference data set, generate 30 simulated datasets (of the same size that the reference dataset) with each simulation method.
3. Compute the p-value of the goodness-of-fit test comparing reference and simulated datasets.
4. For each method, there is 900 p-values obtained in the same conditions (same number of variables, same size of the learning set, same correlation degree between variables). Then plot the boxplot of the obtained p-values.

3.2 Experiment results

3.2.1 Number of variables

We choose to test the effect of the number of variables on the simulation process by varying the number of variables between 3 and 6. The other parameters are unchanged and have been fixed at 30 for the size of the reference datasets, and 0.5 for the correlation coefficient between criteria. The results are shown in Figure 2. One can see that the effect of changing the number of variables is pretty null, as the four boxplots are very similar. The three methods have almost the same performance whatever the number of variables is.

3.2.2 Size of the learning set

We choose to test the effect of the learning set size on the simulation process by varying the learning set size between 20 and 50 items. The other parameters are unchanged and have been fixed at 4 for the number of variables, and 0.5 for the correlation coefficient between criteria. The results are shown in Figure 3. One can see that the effect of changing the size of the learning set is different for the three methods. The effect is null for the copula method. The effect is small for the PCA method and more important for the independent method : results are worse with an increase of the size of the learning set, certainly because an increase of the size of the datasets has a direct effect on the power of the test, i.e. the capacity if the test to reject H_0 when H_0 is false.

3.2.3 Variables correlations

We choose to test the effect of the correlation degree on the simulation process by varying the correlation degree between 0.1 and 0.9s. The other parameters are unchanged and have been fixed at 4 for the number of variables, and 30 for the size of the learning set. The results are shown in Figure 4. One can observe the differences between the three methods with the correlation degree between the variables. Just remember that a post-treatment has been done to modify the data in order to have a functional (but not linear) link between the variables. The results are very interesting: one can easily see that the three methods have also the same results when the correlation degree is weak, but when the correlation degree is strong, the simulation method based on copula is able to capture the link between variables. The PCA method is also able to capture this link, less powerful in the case of non-linear link. The independent method, as guessed, does not simulate data similar to the initial ones when there is a correlation between variables. Therefore, it is clear on the Figure 4 that the simulation method based on PAC or copula produces simulated data that are more similar to a set of initial data than independent samples.

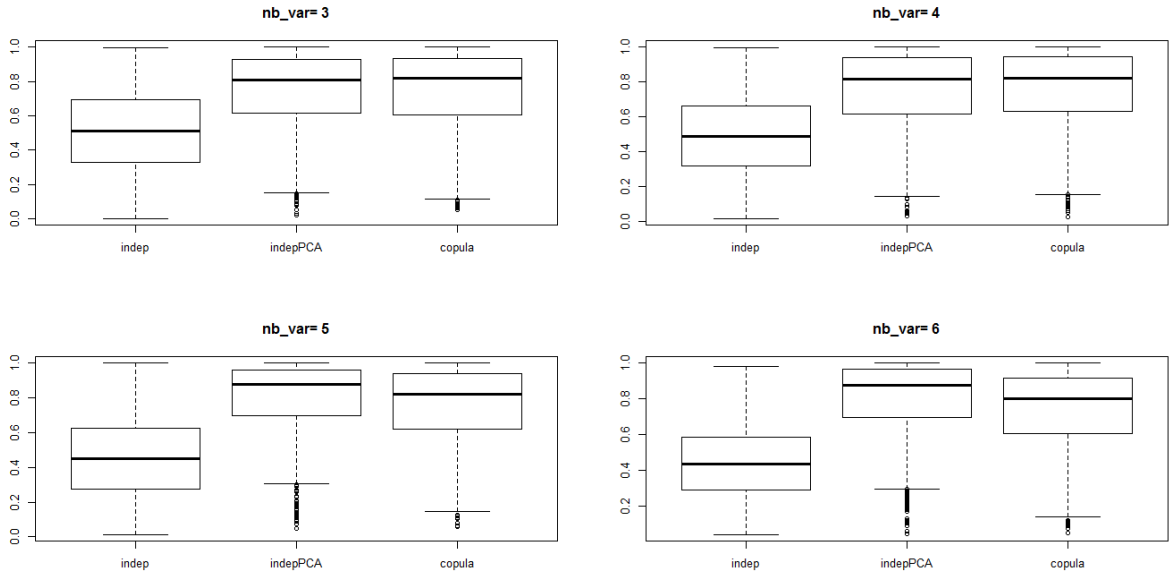


Figure 2: Variation of the quality plots with the number of variables – number of variables=3, 4, 5, 6

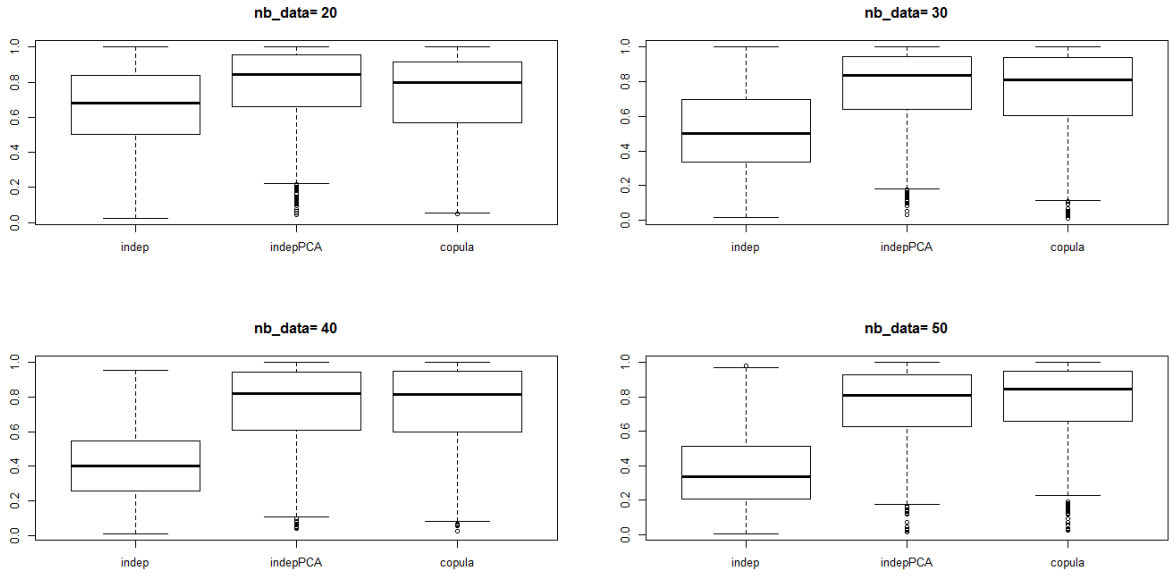


Figure 3: Variation of the quality plots with the size of the learning set – number of learning items= 20, 30, 40, 50

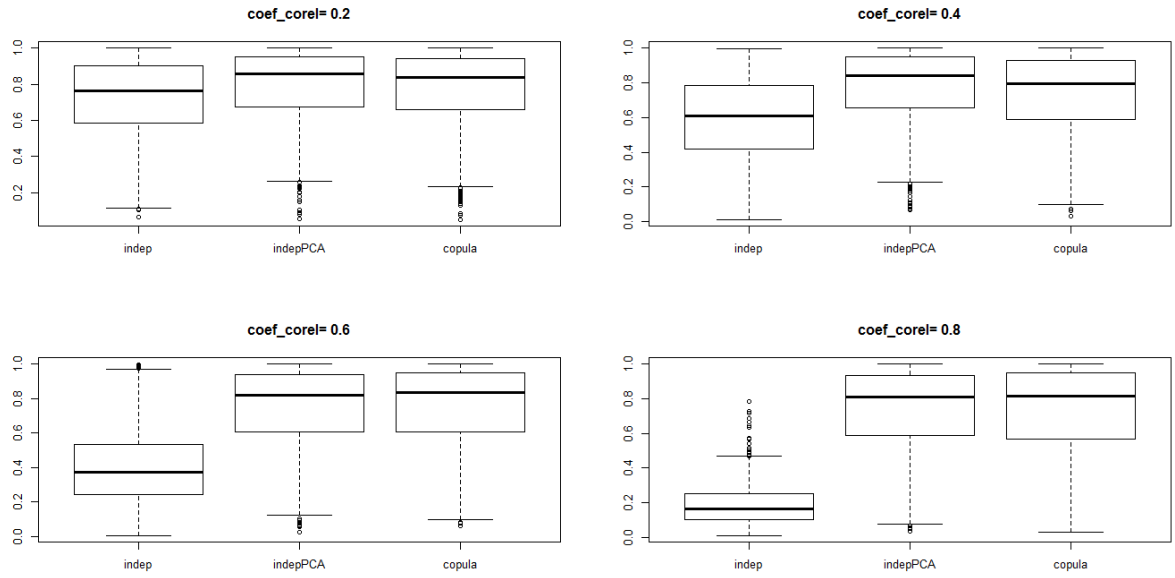


Figure 4: Variation of the quality plots with the correlation degree – cor. coef= 0.2, 0.4, 0.6, 0.8

3.3 Results on real datasets

We propose in this section to test the three different methods on some real datasets from the literature. The first one, proposed in [1], has 4 variables and 29 observations. The variables are almost independent. The second one is introduced in [2], and has 14 observations for 5 criteria. The correlation index between variables are around 0.3 The last one was proposed in [3] and has 27 observations for 4 variables. The correlation coefficient between variables are around 0.7. For each dataset, we produced 500 simulated datasets and then we draw the p-value boxplot as before. Results are shown in Figures 5a, 5b and 5c.

As a result, we can observe that the PCA sample method seems to produce more accurate simulated data, even if the copula sample method leads also to good quality simulated data. For the first dataset, the three different methods are similar, even if the independent sample method do not produce the same quality datasets as the two others. It seems that even if no correlation is observed between the variables in the initial dataset, PCA and copulas sample methods are able to catch a small dependent link and therefore lead to more accurate data generation. For the second data set also the three methods seems to produce sampling data very close to the initial ones. PCA sampling method seems also in this case be the best method, i.e. the method that produces new data that are impossible to discriminate from the initial ones via a goodness-of-fit test. The third case is different, as it is very clear in this case that the independent sampling method is not efficient : Figure 5c shows that most of the sampling produced by the independent methods can be distinguished from the initial dataset, whereas those produced with the copula sampling method, or even better with the PCA sampling method, can be considered as similar to the initial dataset.

These three examples give a good illustration that taking into account dependencies between variables (even if there are not obvious) leads to better simulated data than just independent sampling method. However, it is a surprise for us that on these three examples PCA sampling method seems to produce better results than copula-based method.

4 Conclusion

We proposed two different ways to improve the simulation quality of generated datasets for Multicriteria Decision Process. Both copulas and PCA methods lead to simulated datasets that are more accurate than pure independent samples. Copula method seems to have better results within controlled environment, whereas on the 3 “real” tested datasets PCA method seems to produce better results. However, we strongly encourage practitioners to use one of these two methods to extend the datasets they’re working on and then generate new datasets to test the proposed decision process.

REFERENCES

[1] R. Botreau and A. Rolland. Evaluation multicritère du bien-être animal en ferme : une application des méthodes développées en aide à la décision multicritère. In *8eme congrès de la ROADEF, Clermont-Ferrand*, 2008.

[2] D. Bouyssou, T. Marchant, M. Pirlot, P. Perny, A. Tsoukiàs, and Ph. Vincke. *Evaluation and decision models: a critical perspective*. Kluwer Academic, Dordrecht, 2000.

[3] Th. Briggs, P.L. Kunsch, and B. Mareschal. Nuclear waste management: An application of the multicriteria promethee methods. *European Journal of Operational Research*, 44(1):1 – 10, 1990.

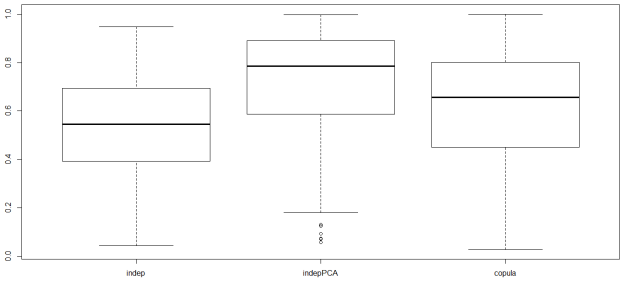
[4] J. Cugliari, A. Rolland, and T.M.T. Tran. On the use of copulas to simulate multicriteria data. In *DA2PL 2014 Workshop (proceedings)*, pages 3–9, 2014.

[5] Michael P. McAssey. An empirical goodness-of-fit test for multivariate distributions. *Journal of Applied Statistics*, 5(40):1120–1131, 2013.

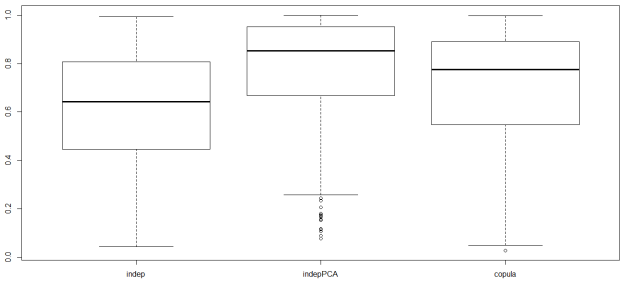
[6] R.B. Nelsen. *An Introduction to Copulas*. Springer, second edition, 2006.

[7] B. Roy. *Multicriteria Methodology for Decision Aiding*. Kluwer Academic Publisher, 1996.

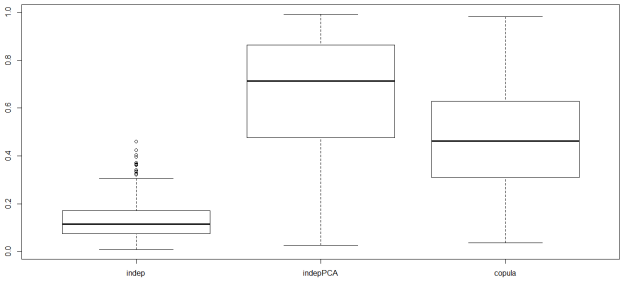
[8] Gabor J. Szkely and Maria L. Rizzo. Testing for equal distributions in high dimension. *InterStat*, (5), 2004.



(a) Case 1: Botreau and Rolland



(b) Case 2: Bouyssou *et al.*



(c) Case 3: Briggs *et al.*

Figure 5: Real datasets

Optimal Subset Selection With Pairwise Comparisons

Matthew Groves[1] and **Juergen Branke** [2]

Abstract. In this work, we study an adaptation of the subset selection problem where information about alternatives can only be collected through pairwise comparisons. We adapt two well known Bayesian sequential sampling techniques, the Knowledge Gradient policy and the Optimal Computing Budget Allocation framework for the pairwise setting and compare their performance on a range of empirical tests. We demonstrate that these methods are able to match or outperform the current state of the art racing algorithm approach.

1 Introduction

In this work, we consider an adaptation of the classic optimal subset selection problem. The aim is to efficiently and accurately select the best subset of given size from a set of possibilities, where the "quality" of each alternative has to be estimated through a noisy sampling process.

In the standard problem, the score of each of K possible options is modelled by the expectation of a real-valued random variable, a statistic estimated through repeated sampling. The total number of samples available is generally restricted, giving rise to an optimisation problem, with the objective of devising sampling procedures to maximise the probability of correctly selecting the highest scoring option or subset of options within the sampling budget constraint.

The adaptation we consider restricts the sampling process to allow only pairwise comparisons between alternatives. Here, rather than modelling the score of an option as a random variable that can be sampled directly, we instead treat the outcome of each possible pairwise comparison between options as a random variable. The score of an option is therefore considered to be the sum of the expectation of the $K-1$ R.V.'s for the pairwise comparisons it is involved in.

This sampling restriction increases the complexity of the problem, with the number of individual samples required to obtain a single measurement of the score of all possible options increasing from K to $\frac{K(K-1)}{2}$. In addition, the outcomes of the pairwise comparisons need not be transitive, and the information gained from a particular pairwise comparison only relates to part of an option's overall quality, leading to additional complications when attempting to optimise the sampling process.

In the following section, we give a brief overview of current approaches both to the standard and pairwise subset selection problems. Section 3 describes the problem in more detail before an overview of two different sampling strategies that can be adapted to the pairwise problem in Section 4. We then apply each of these strategies to various problem settings, with the results of empirical testing discussed in Section 5. We conclude in Section 6.

[1] Mathematics Department, University of Warwick, email: M.J.Groves@warwick.ac.uk

[2] Warwick Business School, University of Warwick, email: Juergen.Branke@wbs.ac.uk

2 Related Work

One well known approach for ranking and selection problems is through the use of racing algorithms, first introduced in [16] and [17]. These iterative methods provide a framework for dealing with sampling uncertainty, designed to replicate a race. During the sampling process, as the quantity of information about each alternative increases, particularly well-performing options can be allowed to "finish early" and are selected, while those that lag behind are eliminated, with further sampling focussed solely on alternatives remaining in the race. Although the standard racing implementation includes the idea of a maximum budget, the proportion of this budget utilized by the race is usually variable, as the algorithm terminates upon reaching some given level $(1-\delta)$ of accuracy based on a probabilistic bound. Fixed budget adaptations of the racing framework (see for example [2]) do exist, aiming to adaptively tune the accuracy parameter δ to maximise performance within a given budget constraint.

Racing has been applied to a variety of contexts including model selection and parameter tuning. [14] describe in detail the *selectRace* procedure for obtaining the best μ of λ options, using both the Hoeffding and empirical Bernstein bounds. [5] and [3] adapt the Hoeffding racing algorithm to the pairwise preference sampling problem, proposing a preference-based racing (PBR) framework with three different sampling strategies. They compare these strategies using real-world football data, and empirically demonstrate their performance on several synthetic "top-κ" selection problems. To our knowledge, this work represents the current state of the art for subset selection using pairwise comparisons with sampling uncertainty. For a good overview of pairwise comparisons in other ranking contexts, see [4].

Several alternative approaches are discussed in [1]. In particular, the authors describe in detail two classes of Bayesian methods; expected value of information procedures (VIP) and the optimal computing budget allocation (OCBA).

OCBA refers to a group of procedures first proposed in [6], and further developed in [8] and [9]. In [7], the authors adapt a version of OCBA for optimal subset selection. The problem considered in this work is the classic selection problem, where allocating a simulation run corresponds directly with sampling the score of an alternative. However, as OCBA is well suited to stochastic simulation optimization problems, and has been shown to be an effective sampling policy for subset selection, it seems reasonable to expect OCBA could be well applied to the pairwise problem.

A popular variant of the VIP approach is the Knowledge Gradient (KG) policy first proposed in [13] and developed in [12] and [10]. The KG policy sequentially samples alternatives based on myopically optimising the expected value of information gained by performing a single additional sample. [12] demonstrate that the KG policy is able to perform efficiently where sample measurements are

normally distributed. [15] identifies potential limitations of the KG for discrete measurement cases, proposing adapted sample selection methods demonstrated to improve performance in the Bernoulli case. However, like OCBA, KG has so far only been applied to ranking and selection scenarios where the quality of alternatives can be measured directly, rather than with pairwise preference sampling.

3 Problem Description

The problem we consider is a variation of the standard "Top κ Selection" problem. Suppose we are presented with a finite set of K possible options $\mathcal{O} = \{o_1, o_2, o_3, ..., o_K\}$ and to each possible pair of alternatives (o_i, o_j), there is associated a random variable $X_{i,j}$ with unknown finite mean $\mu_{i,j}$, representing the expected outcome of a "pairwise comparison" of option o_i against option o_j. The quality S_i of an alternative o_i is determined by the sum of the means of the R.V.'s corresponding to pairwise comparisons of o_i against other options:

$$S_i = \sum_{j \neq i} \mu_{ij}$$

We assume that comparing an option o_i to o_j has the same effect as comparing o_j to o_i. Thus, the random variables $X_{i,j}$'s are paired, with $X_{i,j} = -X_{j,i}$ and performing a pairwise comparison of two options will affect both of their scores.

The aim is to identify the index set $\mathcal{I} \in [K]$ of given size κ containing the highest scoring alternatives, i.e, the solution to the optimization problem:

$$\underset{\mathcal{I} \subset [K]:|\mathcal{I}|=\kappa}{\operatorname{argmax}} \sum_{i \in \mathcal{I}} S_i$$

This can be done by repeatedly selecting option pairs (o_i, o_j) and sampling $X_{i,j}$, thereby improving the quality of our estimates of the $\mu_{i,j}$'s that comprise the option's scores. In particular, we are interested in cases where the sampling process is deemed "expensive", and hence the maximum number of samples we can take is limited. Given a fixed sampling budget, the problem becomes how to select the sampling pairs to maximise the probability of correctly identifying the optimal subset at the end of the sampling process.

4 Algorithm Details

In our literature review, we identified two possible sampling policies that can be adapted to address the problem above. In this section, we discuss them in more detail, along with our modifications for pairwise sampling.

4.1 OCBA-m

OCBA is an asymptotically optimal budget allocation policy based on a Bayesian framework. Since it was first proposed in [6], several different variants of OCBA have been developed ([8], [9]). The procedure was adapted in [7] for optimal subset selection, with the name OCBA-m to refer to the selection of multiple elements.

In this work, we implement the variant described in [1], which we adapt both for selecting a subset rather than a single option and to use pairwise comparisons. At each stage of the sampling process OCBA aims to maximise the estimated increase in the probability of correct selection (PCS) gained from the sample. To do this, it considers the effect of allocating a single additional sample to a particular pairwise comparison and none to the others. The assumption is that, by collecting an additional sample from the random variable corresponding to that pair, the standard error of our estimate of the mean of the outcome from that pairwise comparison will decrease. We model this effect with the approximate assumption of Gaussian distributions over these means. We therefore assume that the allocation of a single sample to the pair (o_i, o_j) scales the distribution of the score $\tilde{\mathcal{S}}_p$ of an option o_p thus:

$$\tilde{\mathcal{S}}_p^{i,j} \sim \mathcal{N}(\hat{\boldsymbol{\mu}}_{\boldsymbol{p}}^{\boldsymbol{i,j}}, \hat{\boldsymbol{\sigma}}_{\boldsymbol{p}}^{\boldsymbol{i,j}}) \sim \mathcal{N}\Bigg(\sum_{q,q \neq p} \tilde{\mu}_{p,q}, \sum_{q,q \neq p} \frac{\tilde{\sigma}_{p,q}^2}{n_{p,q} + \mathbb{I}\{p,q = i,j\}} \Bigg)$$

where $n_{p,q}$ denotes the number of samples performed of a pair (o_p, o_q), with $\tilde{\mu}_{p,q}$ and $\tilde{\sigma}_{p,q}^2$ the corresponding sample mean and sample variance, and where $\mathbb{I}\{p,q = i,j\}$ is the indicator function that returns 1 if either $p = i$ and $q = j$, or $q = i$ and $p = j$. Calculating the $\tilde{\mathcal{S}}_p^{i,j}$'s allows us to obtain an estimate for the probability of correct selection after having performed the additional comparison of (o_i, o_j):

For a constant c,

$$\begin{aligned} EPCS_{i,j} &= \mathbb{P}\{\tilde{\mathcal{S}}_p^{i,j} > \tilde{\mathcal{S}}_q^{i,j} \text{ , for all } p \in \mathcal{I}, q \notin \mathcal{I}\} \\ &\approx \prod_{p \in \mathcal{I}, q \notin \mathcal{I}} \mathbb{P}\{\tilde{\mathcal{S}}_p^{i,j} > \tilde{\mathcal{S}}_q^{i,j}\} \text{ (Assume independence)} \\ &\geq \prod_{p \in \mathcal{I}} \mathbb{P}\{\tilde{\mathcal{S}}_p^{i,j} > c\} \prod_{q \notin \mathcal{I}} \mathbb{P}\{\tilde{\mathcal{S}}_q^{i,j} < c\} \equiv AEPCS_{i,j} \end{aligned}$$

As we only need the relative values of the $EPCS_{i,j}$'s, we use the lower bound *approximate expected probability of correct selection* ($AEPCS_{i,j}$) as described in [7]. We set:

$$c = \frac{\hat{\boldsymbol{\mu}}_{\boldsymbol{\kappa}}^{\boldsymbol{i,j}} + \hat{\boldsymbol{\mu}}_{\boldsymbol{\kappa+1}}^{\boldsymbol{i,j}}}{2}$$

where o_κ and $o_{\kappa+1}$ are the options currently ranked κ^{th} and $(\kappa + 1)^{\text{th}}$ respectively. At each step, we select the pair that maximises $AEPCS_{i,j}$

4.2 Knowledge Gradient

Knowledge Gradient (KG) is a one-step Bayesian look-ahead policy that aims to maximise the expected value gained by performing an additional sample under the assumption that the sampling process will terminate once the sample has been collected. In the context of optimal subset selection, given an index set $\mathcal{I}$, its value is typically determined by the zero-one loss function:

$$U(\mathcal{I}) = \begin{cases} 1, & \text{if } \mathcal{I} \text{ is correct} \\ 0, & \text{otherwise} \end{cases}$$

Suppose during the sampling process, we currently consider the index set $\mathcal{I}$ to contain the κ best options and denote the (as yet unknown) best index set we would obtain after sampling a pair (o_i, o_j) by $\mathcal{I}^{i,j}$. The expected value gain of performing such a sample is simply:

$$V^{i,j} = \mathbb{P}\{U(\mathcal{I}^{i,j}) = 1 | U(\mathcal{I}) = 0\} - \mathbb{P}\{U(\mathcal{I}^{i,j}) = 0 | U(\mathcal{I}) = 1\}$$

However, as we do not know the value of $U(\mathcal{I})$ during our sampling process, $V^{i,j}$ cannot be computed. To allow us to approximate it, we make the assumption that performing further sampling will not cause

Pairwise OCBAm Procedure

INPUT: Set of K options $\{o_1, .., o_K\}$,
Required selection size κ,
Total sampling budget N_{Total},

INITIALIZE: Perform $n_{initial}$ samples of each pair of options;
$n_{i,j} = n_{initial}$ for all i, j

LOOP: WHILE $\sum_{p,q} n_{p,q} < N_{Total}$ DO:

UPDATE: Sample means $\tilde{\mu}_{p,q} = \frac{1}{n_{p,q}} \sum X_{p,q}$,
Standard dev. $\tilde{\sigma}_{p,q} = \sqrt{\frac{1}{n_{p,q}-1} \sum (X_{p,q} - \tilde{\mu}_{p,q})^2}$,
Option scores $S_p = \sum_{q,p \neq p} \tilde{\mu}_{p,q}$,
Recalculate index set $\mathcal{I}$ of best κ options.

FOR ALL PAIRS (o_i, o_j):

UPDATE: Option score means $\hat{\boldsymbol{\mu}}_{\boldsymbol{p}}^{\boldsymbol{i,j}} = S_p$,
Option score std. devs. $\hat{\boldsymbol{\sigma}}_{\boldsymbol{p}}^{\boldsymbol{i,j}} = \sum_{q,q \neq p} \frac{\tilde{\sigma}_{p,q}^2}{n_{p,q} + \mathbb{I}\{p,q=i,j\}}$
Boundary value $c = \frac{\hat{\boldsymbol{\mu}}_{\kappa}^{\boldsymbol{i,j}} + \hat{\boldsymbol{\mu}}_{\kappa+1}^{\boldsymbol{i,j}}}{2}$
Compute $AEPCS_{ij}$

SELECT: Select pair (o_i, o_j) that maximises $AEPCS$,
Perform sample of (o_i, o_j),
$n_{i,j} \leftarrow n_{i,j} + 1$

END LOOP

RETURN $\mathcal{I}$

us to discard a correct index set $\mathcal{I}$, i.e that $\mathbb{P}\{U(\mathcal{I}^{i,j}) = 0 | U(\mathcal{I}) = 1\} = 0$. Although this is evidently untrue in practice, we assume that the information gained by sampling as we move from $\mathcal{I}$ to $\mathcal{I}^{i,j}$ improves our ability to correctly identify the correct index set, making:

$$\mathbb{P}\{U(\mathcal{I}^{i,j}) = 1 | U(\mathcal{I}) = 0\} >> \mathbb{P}\{U(\mathcal{I}^{i,j}) = 0 | U(\mathcal{I}) = 1\}$$

With this assumption, we define the approximate value gain of sample $AV^{i,j}$ as:

$$\begin{aligned} AV^{i,j} &= \mathbb{P}\{U(\mathcal{I}^{i,j}) = 1 | U(\mathcal{I}) = 0\} \\ &\approx \mathbb{P}\{\mathcal{I}^{i,j} \neq \mathcal{I}\} \end{aligned}$$

To calculate $AV^{i,j}$, we first compute the increase in estimated score for options o_i and o_j that would be required to alter $\mathcal{I}$, which we call $\delta_i^{i,j}$ and $\delta_j^{i,j}$. There are several cases for calculating $\delta_i^{i,j}$ and $\delta_j^{i,j}$, dependent on whether o_i and o_j are present in the current estimated index set $\mathcal{I}$:

$$\delta_i^{i,j} = \begin{cases} S_\kappa - S_i & \text{if } o_i, o_j \notin \mathcal{I} \\ S_j - S_{\kappa+1} & \text{if } o_i, o_j \in \mathcal{I} \\ min\{\frac{S_j - S_i}{2}, S_\kappa - S_i, S_j - S_{\kappa+1}\} & \text{if } o_j \in \mathcal{I}, o_i \notin \mathcal{I} \\ \infty & \text{if } o_i \in \mathcal{I}, o_j \notin \mathcal{I} \end{cases}$$

and vice versa for $\delta_j^{i,j}$. Once δ_i has been calculated, the sampling outcome required to affect a change of size δ_i in the score of option o_i is simply:

$$\Delta_i^{i,j} = \delta_i^{i,j}(n_{i,j} + 1) + \tilde{\mu}_{i,j}$$

which we use to obtain $AV^{i,j}$:

$$AV^{i,j} = 2 - \left[\Phi\left(\frac{|\Delta_i^{i,j} - \tilde{\mu}_{i,j}|}{\tilde{\sigma}_{i,j}} \right) + \Phi\left(\frac{|\Delta_j^{i,j} - \tilde{\mu}_{i,j}|}{\tilde{\sigma}_{i,j}} \right) \right]$$

Where Φ is the cumulative distribution function of the standard normal distribution. At each step, we choose to perform the sample that maximises AV.

Knowledge Gradient Procedure (Normally dist. measurements)

INPUT: Set of K options $\{o_1, .., o_K\}$,
Required selection size κ,
Total sampling budget N,

INITIALIZE: Perform n_0 samples of each pair of options;
$n_{i,j} = n_0$ for all i, j

LOOP: WHILE $\sum_{p,q} n_{p,q} < N$ DO:

UPDATE: Sample means $\tilde{\mu}_{p,q} = \frac{1}{n_{p,q}} \sum X_{p,q}$,
Standard dev. $\tilde{\sigma}_{p,q} = \sqrt{\frac{1}{n_{p,q}-1} \sum (X_{p,q} - \tilde{\mu}_{p,q})^2}$,
Option scores $S_p = \sum_{q,p \neq p} \tilde{\mu}_{p,q}$,
Recalculate index set $\mathcal{I}$ of best κ options.

FOR ALL PAIRS (o_i, o_j):

UPDATE: Required score changes $\delta_i^{i,j}$ and $\delta_j^{i,j}$
Compute $\Delta_i^{i,j}$,$\Delta_j^{i,j}$ and $AV^{i,j}$

SELECT: Select pair (o_i, o_j) that maximises $AV^{i,j}$,
Perform sample of (o_i, o_j),
$n_{i,j} \leftarrow n_{i,j} + 1$

END LOOP

RETURN $\mathcal{I}$

5 Test Scenarios

We consider 2 different test scenarios, each a variation of the same simple model, but with significant differences in the outcomes of the pairwise samples. In the first scenario, we model situations where the preference magnitude of one option over another in a comparison is meaningful, with the results of pairwise comparisons unbounded and continuous. In the second scenario outcomes are binary, modelling situations where comparing options provides a simple win/loss outcome. Both scenarios provide interesting difficulties for our sampling algorithms.

Throughout, we assume the existence of a hidden vector $T \in \mathbb{R}^K$ that contains the "true" quality values of each of the K options. In all of our tests, T is randomly generated, with each element $t_i \sim U[0, k]$, where k is a parameter designed to represent the "difficulty" of the problem. As k decreases, the expected difference in the underlying quality values of the different options decreases, making it more difficult to distinguish between them.

The distributions of the pairwise random variables (the $X_{i,j}$'s) are dependent on the corresponding underlying quality values in T. For a pair of options (o_i, o_j), we set $X_{i,j} \sim N(t_i - t_j, 0.5)$. In our first scenario, we can sample directly from the $X_{i,j}$, and our estimate of the quality of each alternative is the sum of our estimates of the means. In scenario 2, with binary responses to pairwise comparisons, we instead sample from proxy random (Bernoulli) variables $Y_{i,j}$, defined thus:

$$Y_{i,j} = \begin{cases} 1, & \text{for } X_{i,j} \geq 0 \\ 0, & \text{for } X_{i,j} < 0 \end{cases}$$

For each of the 2 scenarios, we performed 10 different instances, varying the difficulty parameter k from 1 (hardest) to 10 (easiest). In all cases, we used system size $K = 10$ and subset size $\kappa = 5$.

We compare the performance of the proposed sampling policies against a benchmark racing algorithm, replicating the Preference-Based Racing (PBR) algorithm in [5]. We also compare against a random sample selection policy. For comparison against the racing algorithm, the sampling budget for the other policies was determined by the average stopping time of the racing algorithm. In all other comparisons, we used a sampling budget of 1000.

5.1 Scenario 1: Unbounded, Gaussian

In the case of pairwise comparisons with unbounded outcomes, we encountered difficulties with the Hoeffding bound-based PBR policy. The Hoeffding bound requires the support of the sampling distribution to be strictly bounded, which is untrue in this case. This issue is acknowledged in [5], where they propose introducing an approximate range bound, wide enough to hold with high probability for the number of samples that will be taken. They demonstrate that this approach can perform well in cases where the score of an alternative can be directly sampled. However, we would require approximating a range bound for each pairwise random variable $X_{i,j}$. This greatly increases the potential for error, as we now require $\frac{K^2-K}{2}$ additional approximations. Furthermore, the overall score of an option is estimated by the sum of the estimates of the means of all the relevant pairwise comparisons. It therefore requires a far greater number of samples of each pair to sufficiently tighten the confidence intervals on option scores to allow us to eliminate any from the race.

To address this, we replace the Hoeffding bound in this instance with a T-statistic confidence interval. This incurs some loss of generality, restricting this racing method to normally distributed samples, whereas the Hoeffding bound should in principle hold for any bounded distribution.

Figure 1 shows the relative performance of the different policies for the 10 different instances of scenario 1 ($k = \{1, 2, ..., 10\}$), with sampling budget determined by the average completion time of the T-statistic racing algorithm. Results shown are an average of 1000 replications with different random seeds. We observe that for the easiest problems ($k = 10, 9, 8$), all of the policies perform well, as even random allocation of the sampling budget is enough to distinguish between such widely spaced options. As the difficulty parameter increases, the success rate for all of the methods decreases, even thought the number of samples available increases. At first, the racing algorithm performs well, comparable to the KG policy, with the OCBA-m policy 2-4% worse, whilst the performance of the random allocation policy decreases sharply. For the most difficult problems, a clear separation in the success rates can be observed, with both KG and OCBA-m performing significantly better than the PBR.

Figure 2 compares KG and OCBA-m against random allocation for a sampling budget of 1000 at each k value. Both methods perform significantly better than random allocation at all difficulty levels, with similar success rates for medium and high k values. For the more difficult problems, we see again that KG is able to outperform OCBA-m by a margin of approximately 2-5%.

Finally, we investigated the performance of KG and OCBA-m for different simulation budgets using a single value for k, choosing $k = 4$ as a medium difficulty problem for which the success rates of the two policies were very similar when compared to the PBR. Figure 3 shows the success rates as the simulation budget increases from 100 to 2000 samples. We see that, for this k value, KG is consistently 1-2% better than OCBA-m, although the margin of improvement is within the standard error of the measurement for much of the sampling range.

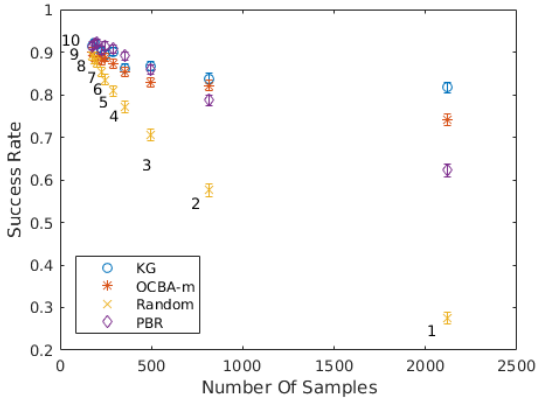


Figure 1. Performance of the KG, OCBA-m and PBR (t-bound) against random allocation for $k = \{1, 2, .., 10\}$ in Scenario 1. Sampling budget determined by the average stopping time of the PBR

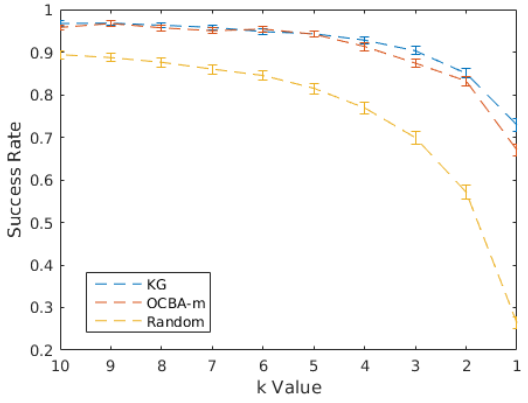


Figure 2. Performance of the KG, OCBA-m and random allocation policies for a fixed sampling budget of 1000 for $k = \{1, 2, .., 10\}$ in Scenario 1.

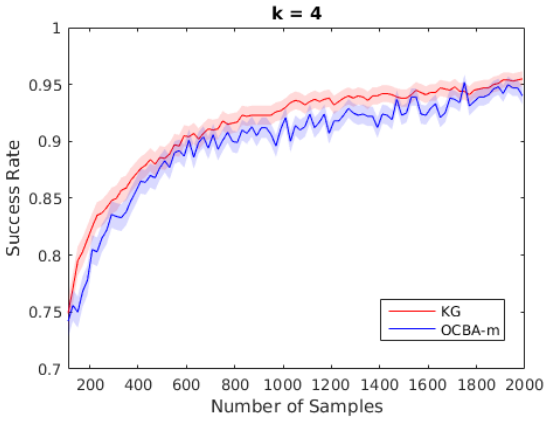


Figure 3. Comparison of the performance of the KG and OCBA-m as the sampling budget increases for $k = 4$ in Scenario 1.

5.2 Scenario 2: Bounded, Bernoulli

Introducing the Bernoulli proxy random variables changes the scenario in several significant ways. Firstly, as results of pairwise comparisons are now restricted to either 0 or 1, the Hoeffding bound condition for the PBR is again suitable, and sufficiently tight to terminate before reaching N.

Secondly, bounded sample results limit the effectiveness of the knowledge gradient policy. At each step of this algorithm, we calculate an estimate for each pair (o_i, o_j), of the probability that collecting a single additional sample will change the scores of options i and j sufficiently to alter our top-rated subset. However, restricting the results of the sample to binary values limits the change that sampling can make to the alternative's scores. Specifically, suppose we are in some knowledge state θ, with an estimate of $\tilde{\mu}^{\theta}_{i,j}$ for $Y_{i,j}$ (and thus $\tilde{\mu}^{\theta}_{j,i} = 1 - \tilde{\mu}^{\theta}_{i,j}$ for $Y_{j,i}$) and we perform a single additional sample of $Y_{i,j}$. Then we will have:

$$\tilde{\mu}^{\theta+1}_{i,j} = \begin{cases} \tilde{\mu}^{\theta}_{i,j} + \frac{1}{n_{i,j}+1}(1 - \tilde{\mu}^{\theta}_{i,j}), & \text{if } Y_{i,j} = 1 \\ \tilde{\mu}^{\theta}_{i,j} - \frac{\tilde{\mu}^{\theta}_{i,j}}{n_{i,j}+1}, & \text{if } Y_{i,j} = 0 \end{cases}$$

and vice versa for $\tilde{\mu}^{\theta}_{j,i}$. If it is the case that the required difference in estimated score to cause a change for all pairs of options exceeds these possible change amounts, then no single pairwise sample will be able to alter the current top set. This means that our knowledge gradient values will be:

$$AV^{ij} = 0, \forall (o_i, o_j)$$

and our Knowledge Gradient policy will be unable to select a sample. This potential problem with the KG policy was hinted at in [18], and discussed in detail in [15].

It becomes necessary to consider multiple samples in order to find sampling sequences with positive change probabilities. [11] propose the adapted KG(*) policy, which considers sequences of samples, selecting to perform the sample at the start of the shortest sequence required to change the ranking. They show that this policy performs well, but can be very computationally intensive, as the state space of sampling sequences grows rapidly as the sequences lengthen. In the case of pairwise sampling, with $\frac{1}{2}(K^2 - K)$ possible sample pairs at each stage, this method rapidly becomes computationally intractable for even modest values of K.

To solve this, [15] suggests an alternative method for formulating $AV^{i,j}$, leading to an adapted policy KG(min), that allocates based on minimising the number of consecutive repeated samples of a single alternative needed to change the selection. For the standard subset selection problem, where simulation directly estimates the score of alternatives, this is sufficient to prevent the policy from failing, as for any possible option score value S and accuracy ϵ, there is a finite string of sampling outcomes that could can move the score estimate of an option to within ϵ of S, with non-zero probability. However, this is not necessarily true in the pairwise problem. Let $r_{i,j}$ denote the minimum number of consecutive samples of the pair (o_i, o_j) required to change the selected subset.

Proposition 1. *For any $K > 4$ it is possible that $r_{i,j} = \infty$ for all (o_i, o_j) regardless of the selected subset size κ*

Proof : To show this, we aim to construct an example sampling situation whereby $r_{i,j} < \infty \implies K \leq 4$. Suppose at some point in our sampling process we have:

$$\tilde{\mu}_{i,j} = \begin{cases} 0.5 & \text{if } i,j \in \mathcal{I} \\ 0.5 & \text{if } i,j \notin \mathcal{I} \\ 1 & \text{if } i \in \mathcal{I}, j \notin \mathcal{I} \\ 0 & \text{if } i \notin \mathcal{I}, j \in \mathcal{I} \end{cases}$$

then the estimated difference in score of the κ^{th} and $(\kappa+1)^{th}$ best options will be:

$$\begin{aligned} S_\kappa - S_{\kappa+1} &= \sum_{j, j \neq \kappa} \tilde{\mu}_{\kappa,j} - \sum_{j, j \neq \kappa+1} \tilde{\mu}_{\kappa+1,j} \\ &= (0.5(\kappa - 1) + K - \kappa) - 0.5(K - \kappa - 1) \\ &= 0.5K \end{aligned}$$

Now, $\min_{i,j}\{\delta^{i,j}_i\} = \delta^{\kappa+1,\kappa}_{\kappa+1} = \frac{1}{2}(S_\kappa - S_{\kappa+1}) = \frac{K}{4}$, so finitely many samples must be able to alter $\tilde{\mu}_{\kappa+1,\kappa}$ by at least $\frac{K}{4}$ for $r_{\kappa,\kappa+1}$ to be finite. As $\tilde{\mu}_{i,j} \in [0,1]$ for all o_i, o_j, the maximum change in $\tilde{\mu}_{\kappa+1,\kappa}$ we can obtain is 1. Hence, we require:

$$1 \geq \frac{K}{4}$$

■

Proposition 1 means that, when we must use Bernoulli pairwise sampling, and have more than 4 options to choose from, it can be the case that infinitely many samples of a particular pair cannot change the ranking. This means that the KG(min) procedure proposed in [15] can fail. However, this failure might be sufficiently rare for KG(min) to still be a useful selection method in practice. To test this, we performed 10,000 replications of our scenario for each k value and applied KG(min), counting the number of times the failure state was reached. In these 100,000 total runs, each with a budget of 1000 samples, the fail state was somewhat rare, being encountered a total of 892 times, with the highest number of failures for a particular k value, ($k = 4$), of 172. As such, we would tentatively suggest that it is indeed reasonable to apply the KG(min) algorithm in such cases, with an adaptation to allow random sampling if multiple lookahead cannot allocate a sample.

Figure 4 compares the performance of KG(min) and OCBA-m against the PBR (Hoeffding bound) and random sample allocation, with sampling budgets determined by the average stopping time of the racing algorithm over 1000 replications. We observe that all three allocation methods are able to outperform random allocation substantially. For the more difficult problems (lower k values), there are clear differences in performance, with KG(min) having the highest success rate, and both KG(min) and OCBA-m significantly outperforming the PBR.

We also examined the performance of KG(min) and OCBA-m in Scenario 2 when given a sampling budget of 1000 comparisons, with the results shown in Figure 5. Here we see that, with this increased budget, OCBA-m achieves a higher success rate for low problem difficulties, perhaps suggesting better performance for this policy against KG(min) when given a high sampling budget relative to the difficulty of the problem.

To test this, we compared the performance of KG(min) and OCBA-m for a range of different simulation budgets for a single k value, again using $k = 4$, with the results shown in Figure 6. For low sampling budgets, KG(min) performs significantly better than OCBA-m, but as the sampling budget increases, the improvement in success rate for KG(min) slows substantially. As a result, OCBA-m is clearly better for higher sampling budgets, with a 4.7% higher success rate at 2000 samples.

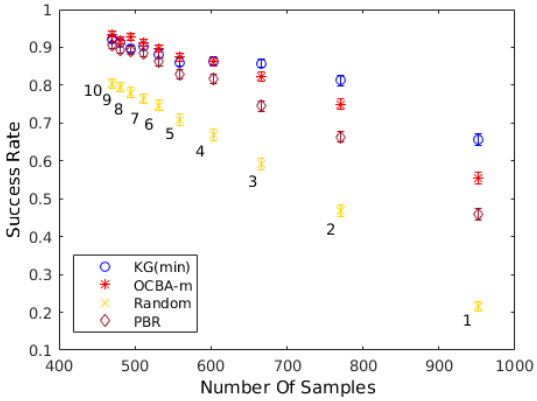


Figure 4. Performance in Scenario 2 of the KG and OCBA-m policies against the PBR (t-bound) and random allocation for $k = \{1, 2, .., 10\}$ (adjacent to lowest data point). Number of samples in each scenario determined by the average stopping time of the PBR.

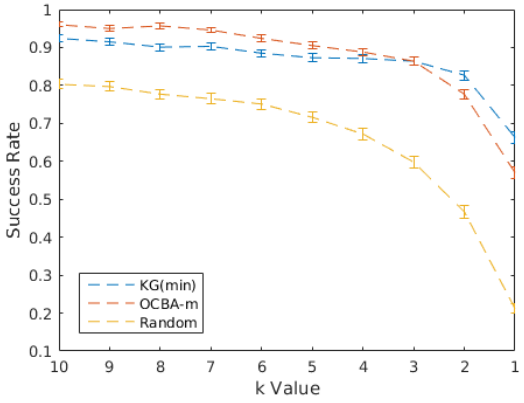


Figure 5. Comparison of the performance of the KG, OCBA-m and random allocation policies for a fixed sampling budget of 1000 for $k = \{1, 2, .., 10\}$ in Scenario 2.

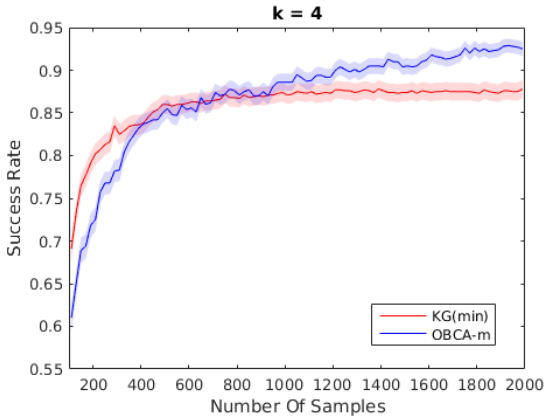


Figure 6. Comparison of the performance of the KG and OCBA-m as the sampling budget increases for $k = 4$ in Scenario 2.

6 Concluding Remarks

In this work we presented two pairwise sampling procedures for optimal subset selection and tested their performance empirically in different synthetic scenarios. Both were able to outperform the state of the art PBR policy on both scenarios. All comparisons of our policies have been simulation-based, so robust theoretical investigation of their properties could be worthwhile.

Additionally, our investigation of scenarios with bounded sampling outcomes identified situations where both the one-step and n-step KG can fail, due to complications idiosyncratic to the pairwise problem. Future work could investigate how occurrences of this failure state are related to both system size and sampling budget.

REFERENCES

[1] J. Branke, S. Chick, and C. Schmidt. Selecting a selection procedure. *Management Science*, 53(12):1916–1932, 2007.

[2] J. Branke and J. Elomari. Racing with a fixed budget and a self-adaptive significance level. In *Learning and Intelligent Optimization*, pages 272–280. Springer, 2013.

[3] R. Busa-Fekete, T. Fober, and E. Hüllermeier. Preference-based evolutionary optimization using generalized racing algorithms. In *23rd Workshop on Computational Intelligence; Dortmund*, pages 237–245. KIT Scientific Publishing, 2013.

[4] R. Busa-Fekete and E. Hüllermeier. A survey of preference-based online learning with bandit algorithms. In *International Conference on Algorithmic Learning Theory*, pages 18–39. Springer, 2014.

[5] R. Busa-Fekete, B. Szorenyi, W. Cheng, P. Weng, and E. Hüllermeier. Top-k selection based on adaptive sampling of noisy preferences. In *30th International Conference on Machine Learning*, pages 1094–1102. JMLR, 2013.

[6] C. H. Chen. A lower bound for the correct subset-selection probability and its application to discrete event simulations. *IEEE Transactions on Automatic Control*, 41(8):1227–1231, 1996.

[7] C. H. Chen, D. He, M. Fu, and L. H. Lee. Efficient simulation budget allocation for selecting an optimal subset. *Informs Journal on Computing*, 20(4):579–595, 2008.

[8] C. H. Chen, J. Lin, E. Yücesan, and S. Chick. Simulation budget allocation for further enhancing the efficiency of ordinal optimization. *Discrete Event Dynamic Systems: Theory and Applications*, 10(3):251–270, 2000.

[9] C. H. Chen, E. Yücesan, L. Dai, and H. C. Chen. Efficient computation of optimal budget allocation for discrete event simulation experiment. *IIE Transactions*, 42(1):60–70, 2010.

[10] S. Chick, J. Branke, and C. Schmidt. Sequential sampling to myopically maximize the expected value of information. *Informs Journal on Computing*, 22(1):71–80, 2010.

[11] P. I. Frazier and W. B. Powell. Paradoxes in learning and the marginal value of information. *Decision Analysis*, 7(4):378–403, 2010.

[12] P. I. Frazier, W. B. Powell, and S. Dayanik. A knowledge-gradient policy for sequential information collection. *SIAM Journal on Control and Optimization*, 47(5):2410–2439, 2008.

[13] S. S. Gupta and K. J. Miescke. Bayesian look ahead one stage sampling allocations for selecting the largest normal mean. *Statistical Papers*, 35(1):169–177, 1994.

[14] V. Heidrich-Meisner and C. Igel. Hoeffding and bernstein races for selecting policies in evolutionary direct policy search. In *26th International Conference on Machine Learning*, pages 401–408. ACM, 2009.

[15] B. Kamiński. Refined knowledge-gradient policy for learning probabilities. *Operations Research Letters*, 43(2):143–147, 2015.

[16] O. Maron and A. Moore. Hoeffding races: accelerating model selection search for classification and function approximation. *Advances in Neural Information Processing Systems*, pages 59–66, 1994.

[17] O. Maron and A. Moore. The racing algorithm: Model selection for lazy learners. *Artificial Intelligence Review*, 5(1):193–225, 1997.

[18] W. B. Powell and I. O. Ryzhov. *Optimal learning*, volume 841. John Wiley & Sons, 2012.

Computing linear rankings from trillions of pairwise outranking situations

Raymond Bisdorff[1]

Abstract. We present in this paper a sparse HPC implementation for outranking digraphs of huge orders, up to several millions of decision alternatives. The proposed outranking digraph model is based on a quantiles equivalence class decomposition of the underlying multicriteria performance tableau. When locally ranking each of these ordered components, we may readily obtain an overall linear ranking of big sets of decision alternatives. For the local rankings, both, Copeland's as well as the Net-Flows ranking rules, appear to give the best compromise between, on the one side, the fitness of the overall ranking with respect to the given global outranking relation and, on the other side, computational tractability for very big outranking digraphs modelling up to several trillions of pairwise outranking situations.

1 Pre-ranked sparse outranking digraphs

When the preference learning community seams now well armed for tackling big preferential data [1], the multiple criteria decision aid community, and especially the one based on the outranking approach, still essentially tackles decision aid problems concerning only tiny or small sets of decision alternatives [2].

Challenged by the computational performances of the preference learning algorithms, we present in this paper a *sparse approximate model* of a given outranking digraph, which allows us, via HPC facilities, to efficiently rank very big sets of decision alternatives. Our starting point is the methodological consideration that an outranking digraph –the association of a set of decision alternatives and an outranking relation (see [3])– is, following the methodological requirements of the outranking based decision aid approach (see [4], [5], necessarily associated with a corresponding *multiple criteria performance tableau*. Such a tableau shows a given set of potential decision alternatives that are evaluated on a given *coherent family* (see Roy [3]) of weighted performance criteria.

In this paper we are going to use the preferential information delivered by such a given performance tableau[2] for linearly decomposing big sets of decision alternatives into ordered quantiles equivalence classes. This decomposition will lead us to a *pre-ranked, sparse approximate model* of an outranking digraph.

[1] University of Luxembourg, Faculty of Science, Technology and Communication, Computer Science and Communications Research Unit/ILIAS, url: http://leopold-loewenheim/bisdorff/ .

[2] Due to space limits we are not going to discuss here the actual elaboration of a performance tableau. We refer instead the reader to our recent book [2].

1.1 Showing the performance tableau

Strong motivation for trying to linearly rank a set of decision alternatives stems from the desire to show the corresponding performance tableau from a decision aiding perspective. Consider, for instance, the performance tableau showing the service quality of 12 commercial cloud providers graded by an external auditor on 14 ordinal, incommensurable performance criteria (see Wagle et al. [6]).

Example 1 (Service quality of commercial cloud providers)**.**

criterion	upT	dwT	ouT	LB	MTBF	Rcv	Lat	RspT	Thrpt	stoC	snpC	auT	enC	auD
Amz	2	2	2	4	3	3	NA	3	NA	4	NA	4	4	4
Cen	4	4	0	4	4	4	NA	2	NA	3	NA	4	4	4
Cit	2	4	2	4	3	4	NA	2	NA	3	4	4	4	4
Dig	2	1	4	4	3	3	NA	2	NA	3	NA	4	4	4
Ela	4	4	0	4	4	4	NA	4	NA	3	4	4	4	4
GMO	1	3	4	4	3	2	NA	4	NA	3	NA	4	4	4
Ggl	4	2	1	4	2	3	NA	2	NA	4	4	4	4	4
HP	3	3	2	4	4	3	NA	4	NA	3	4	4	4	4
Lux	2	2	2	4	3	3	NA	2	NA	2	NA	4	4	4
MS	4	4	0	4	4	4	NA	4	NA	4	NA	4	4	4
Rsp	NA	NA	NA	4	NA	3	NA	NA	NA	3	4	4	4	4
Sig	4	4	0	4	4	4	NA	3	NA	3	4	4	4	4

Legend: *0 = 'very weak', 1 = 'weak', 2 = 'fair', 3 = 'good', 4 = 'very good','NA' = missing data.*

Each row shows the ordinal grades that the auditor has provided for the respective cloud provider. Notice by the way the constant grades on some criteria and the many missing data. It is not evident to discover in this list who might be the potentially best performing cloud provider. The same performance tableau may better be linearly ranked from the best to the worst performing providers; ties, the case given, being resolved lexicographically.

Ranking of cloud providers by service quality

criteria	dwT	Rcv	MTBF	upT	RspT	stoC	auD	enC	auT	snpC	Thrpt	Lat	LB	ouT
weights	2.00	2.00	2.00	2.00	2.00	3.00	1.00	1.00	1.00	3.00	2.00	2.00	2.00	2.00
tau(*)	0.56	0.44	0.44	0.41	0.33	0.14	0.00	0.00	0.00	0.00	0.00	0.00	0.00	-0.45
MS	4	4	4	4	4	4	4	4	4	NA	NA	NA	4	0
Ela	4	4	4	4	4	3	4	4	4	4	NA	NA	4	0
Sig	4	4	4	4	3	3	4	4	4	4	NA	NA	4	0
Cen	4	4	4	4	2	3	4	4	4	NA	NA	NA	4	0
HP	3	3	4	3	4	3	4	4	4	4	NA	NA	4	2
Cit	4	4	3	2	2	3	4	4	4	4	NA	NA	4	2
GMO	3	2	3	1	4	3	4	4	4	NA	NA	NA	4	4
Ggl	2	3	2	4	2	4	4	4	4	4	NA	NA	4	1
Rsp	NA	3	NA	NA	NA	3	4	4	4	4	NA	NA	4	NA
Amz	2	3	3	2	3	4	4	4	4	NA	NA	NA	4	2
Dig	1	3	3	2	2	3	4	4	4	NA	NA	NA	4	4
Lux	2	3	3	2	2	2	4	4	4	NA	NA	NA	4	2

Color legend:

quantile	20.00%	40.00%	60.00%	80.00%	100.00%

(*) tau: Ordinal (Kendall) correlation between marginal criterion and global ranking relation.

The grades observed on each criterion appear optimistically gathered in 5-tile equivalence classes. With 10 grades in the best quintiles class ($80\% - 100\%$), provider 'MS' appears here to be best performing, followed by provider 'Ela'. The criteria usually do not have the

same weight in the decision problem. They appear ranked here in decreasing order of the ordinal correlation index[3] observed between the presentation ranking and the marginal criterion one.

This ranked presentation of a performance tableau is, without doubt, very useful for a decision aiding purpose, even more if the set of decision alternatives becomes bigger. But, how to rank now a big performance tableau gathering the evaluations of thousands or even millions of decision alternatives? The Copeland ranking rule, used for ranking the cloud providers above, is based on net crisp outranking flows which requires to compute the in- and out-degree of each node in the corresponding outranking digraph. When the order n of the outranking digraph now becomes big (several thousand or millions of decision alternatives), this ranking rule will require the handling of a huge set of n^2 pairwise outranking situations.

Yet, it is evident that, when facing such a big set of decision alternatives, the 20% best performing alternatives will for sure outrank the 20% worst performing ones. In a big data case, it may hence appear unnecessary to compute the complete set of the pairwise outranking situations among all decision alternatives. We shall therefore present hereafter a *pre-ranked, sparse approximate model* of the outranking digraph, where we only keep a linearly ranked list of quantiles performance classes with local outranking content.

1.2 Quantiles sorting of a performance tableau

To do so, we propose to decompose the given performance tableau into k ordered *quantiles equivalence classes*. Let X be the set of n potential decision alternatives evaluated on a single real performance criterion. We denote $x, y, ...$ the performances observed of the potential decision alternatives in X. We call *quantile* $q(p)$ the performance such that $p\%$ of the observed n performances in X are less or equal to $q(p)$. The quantile $q(p)$ is determined by the sorted distribution of the observed performances in X.

We consider a series: $p_k = k/q$ for $k = 0, ...q$ of $q+1$ equally spaced quantiles limits like *quartiles* limits: $0, .25, .5, .75, 1$, *quintiles limits*: $0, .2, .4, .6, .8, 1$, or *deciles* limits: $0, .1, .2, ..., .9, 1$. The *upper-closed* q^k *quantiles class* corresponds to the interval $]q(p_{k-1}); q(p_k)]$, for $k = 2, ..., q$, where $p_q = \max_X x$ and the first class $q(p_1) =]-\infty; q(p_1)]$ gathers all data below $q(p_1)$. We call q*-tiles* a complete series of $k = 1, ..., q$ upper-closed q^k quantiles classes. Similarly, the *lower-closed* q_k quantiles class corresponds to the the interval $[q(p_{k-1}); q(p_k)]$, for $k = 1, ..., q-1$, where $p_1 = \min_X x$ and the last class $q(p_q) = [q(p_{k-1}); +\infty[$ gathers all data above $q(p_{k-1})$. We will in the sequel consider by default upper-closed quantiles.

If x is a measured performance on a single criterion, we may hence distinguish three sorting situations: $x \leqslant q(p_{k-1}) \equiv$ 'the performance x is lower than the q^k class'; $x > q(p_{k-1})$ and $x \leqslant q(p_k) \equiv$ 'the performance x belongs to the q^k class'; $x > q(p_k) \equiv$ 'the performance x is higher than the p^k class'. The relation $<$ being the *dual* of $\geqslant$, it will be sufficient to check that both, $q(p_{k-1}) \not\geqslant x$, as well as $q(p_k) \geqslant x$, are verified for x to be a member of the k-th q-tiles class.

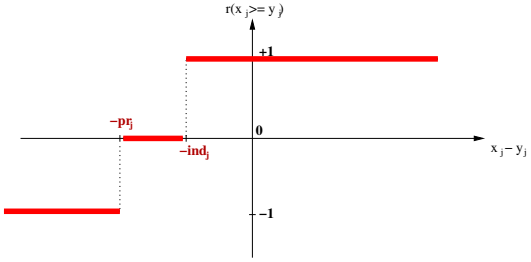
The multiple criteria extension of this single criterion q-sorting works as follows. Let $F = \{1, ..., m\}$ be a finite and coherent family of m performance criteria [4] and let x and y be two evaluation vectors on F. For each criterion j in F, we suppose the performances to be measured on a real scale $[0; M_j]$, supporting an indifference discrimination threshold ind_j, a preference discrimination threshold pr_j and a veto discrimination threshold v_j such that $0 \leqslant ind_j < pr_j < v_j \leqslant M_j$. The marginal evaluation of x on criterion j is denoted x_j. Each criterion j in F carries furthermore a *rational significance* weight w_j such that $0 < w_j < 1.0$ and $\sum_{j\in F} w_j = 1.0$.

In the bipolar outranking approach (see Bisdorff 2013 [9]), every criterion $j \in F$ characterizes on X a marginal double threshold order $\geqslant_j$ with the following semantics (see Fig. 1):

$$r(x \geqslant_j y) \stackrel{\text{def}}{=} \begin{cases} +1 & \text{if} \quad x_j - y_j \geqslant -ind_j \\ -1 & \text{if} \quad x_j - y_j \leqslant -pr_j \\ 0 & \text{otherwise.} \end{cases} \tag{1}$$

$r(x \geqslant_j y) = +1$ signifies that x is *performing at least as good as* y on criterion j, $r(x \geqslant_j y) = -1$ signifies that x is *not performing at least as good as* y on criterion j, and $r(x \geqslant_j y) = 0$ signifies that it is *unclear* whether, on criterion j, x is performing at least as good as y. Each criterion j contributes thus, in the following way, the

Figure 1. Characteristic function of marginal '*at least as good as*' relation



significance w_j of his marginal "*at least as good as*" characterization $r(x \geqslant_j y)$ to the global characterization $r(x \geqslant y)$:

$$r(x \geqslant y) \stackrel{\text{def}}{=} \sum_{j\in F} \left[\, w_j \cdot r(x \geqslant_j y) \,\right], \tag{2}$$

where: $r(x \geqslant y) > 0$ signifies that x is *globally performing at least as good as* y; $r(x \geqslant y) < 0$ signifies that x is *not globally performing at least as good as* y; and, $r(x \geqslant y) = 0$ signifies that it is *unclear* whether x is globally performing at least as good as y.

From an epistemic point of view, we say that evaluation x *outranks* evaluation y, denoted $(x \succsim y)$, if a *significant majority of criteria validates* a global outranking situation between x and y, i.e. $(x \geqslant y)$ and *no veto* (denoted $(x \not\lll y)$) is observed on a discordant criterion. Similarly, evaluation x *does not outrank* evaluation y, denoted $(x \not\succsim y)$, if *a significant majority of criteria invalidates* a global outranking situation between x and y, i.e. $(x \not\geqslant y)$ and *no counter-veto* (denoted $(x \not\ggg_j y)$) is observed on a concordant criterion j [9]. Veto characteristics on a criterion j (denoted $(x \lll_j y)$) are defined as follows:

$$r(x \lll_j y) \stackrel{\text{def}}{=} \begin{cases} +1 & \text{if} \quad x_j - y_j \leqslant -v_j \\ -1 & \text{if} \quad x_j - y_j \geqslant v_j \\ 0 & \text{otherwise.} \end{cases} \tag{3}$$

If we furthermore set $r(x \ggg_j y) \stackrel{\text{def}}{=} r(x \not\lll_j y)$, veto and counter-veto situations will be *codual* one to another.

Now, a global bipolar outranking characteristic $r(x \succsim y)$ is defined as follows:

$$r(x \succsim y) \stackrel{\text{def}}{=} r(x \geqslant y) \;\mathbb{V}\; \left(\mathbb{V}_{j\in F} \left[r(x \not\lll_j y) \right] \right).^5 \tag{4}$$

[3] Extended Kendall τ index, see Bisdorff 2012 [7].

[4] Coherent means here: *universal*, *minimal* and *preferentially consistent wrt marginal preferences* (see Roy 1991, 1993 [3, 8] and Bisdorff 2002 [5]).

In particular: $r(x \succsim y) = r(x \geqslant y)$ if no considerable large positive or negative performance differences are observed; $r(x \succsim y) = 1$ if $r(x \geqslant y) \geqslant 0$ and $\vee_{j \in F}[r(x \ggg_j y)] = 1$; and, $r(x \succsim y) = -1$ if $r(x \geqslant y) \leqslant 0$ and $\vee_{j \in F}[r(x \lll y)] = 1$.

For $k = 1, ..., q$, let $\mathbf{q^k}$ denote a multiple criteria quantiles class, and let $\mathbf{q}(p_{k-1}) = \big(q_1(p_{k-1}), q_2(p_{k-1}), ..., q_m(p_{k-1})\big)$ denote its *lower limits*, and $\mathbf{q}(p_k) = \big(q_1(p_k), q_2(p_k), ..., q_m(p_k)\big)$ its corresponding *upper limits*. Alternative x belongs to multiple criteria quantiles class $\mathbf{q^k}$ if the corresponding lower limits $\mathbf{q}(p_{k-1})$ *do not outrank*, whereas the upper limits $\mathbf{q}(p_k)$ *do outrank* the multiple criteria performances of x. The membership characteristic function evaluating if of alternative x belongs to quantiles class $\mathbf{q^k}$ may be readily assessed as follows:

$$r(x \in \mathbf{q^k}) \stackrel{\text{def}}{=} \min\big[\, -r\big(\mathbf{q}(p_{k-1}) \succsim x\big),\ r\big(\mathbf{q}(p_k) \succsim x\big)\,\big] \quad (5)$$

In our bipolar characteristic calculus, *logical conjunction* is indeed implemented via the min operator, whereas *logical negation* is implemented by changing the sign of the r values (see [4, 5, 9]).

Formula 5 gives us now the essential tool for implementing our q-tiles sorting algorithm.

The multicriteria (upper-closed) q-tiles sorting algorithm

Input: a set X of n decision alternatives with a performance tableau on a family of m criteria and a set $\mathcal{Q} = \{\mathbf{q^1}, ..., \mathbf{q^q}\}$ of $k = 1, .., q$ empty multiple criteria quantiles classes.
For each object $x \in X$ **and each** quantiles class $\mathbf{q^k} \in \mathcal{Q}$

1. $r(x \in \mathbf{q^k}) \quad \leftarrow \quad \min\big(-r(\mathbf{q}(p_{k-1}) \succsim x), r(\mathbf{q}(p_k) \succsim x)\big)$
2. if $r(x \in \mathbf{q^k}) \geqslant 0$ **add** x to q-tiles class $\mathbf{q^k}$

Output: $\mathcal{Q}$

The complexity of the q-tiles sorting algorithm is in $\mathcal{O}(nmq)$, i.e. *linear* in the number of decision alternatives (n), criteria (m) and quantiles classes (q). As $\mathcal{Q}$ represents a partition of the criterion performance measurement scales, there is a potential for run time optimization.

We may compute a didactic example with the help of our DIGRAPH3 collection of Python modules. [6]

Example 2 (Python session with DIGRAPH3 resources)

```
1 >>> from randomPerfTabs import RandomPerformaceTableau
2 >>> t = RandomPerformanceTableau(numberOfActons=50,
  ...                               seed=5)
3 >>> from sparseOutrankingDigraphs import\
  ...                   PreRankedOutrankingDigraph
4 >>> pr = PreRankedOutrankingDigraph(t,quantiles=5)
5 >>> pr.showSorting()
*--- Sorting results in descending order ---*
]0.8 - 1.0]: [a16, a2, a24, a32]
]0.6 - 0.8]: [a01, a02, a06, a09, a10, a13, a16, a18,
              a22, a25, a27, a28, a31, a32, a36, a37,
              a39, a40, a41, a43, a45, a48]
]0.4 - 0.6]: [a01, a03, a04, a05, a07, a08, a09, a10,
              a11, a12, a13, a14, a15, a17, a18, a20,
              a26, a27, a29, a30, a33, a34, a35, a38,
              a42, a43, a44, a45, a46, a47, a49, a50]
]0.2 - 0.4]: [a04, a11, a12, a17, a19, a21, a23,
              a29, a34, a42, a46, a47, a50]
] < - 0.20]: []
```

In Lines 1 and 2 we import a random performance tableau generator class and construct a sample '*RandomPerformanceTableau*' instance called t. In Lines 3 we import the '*PreRankedOutrankingDigraph*' class and in Line 4 we construct a 5-tiles sorting digraph called pr. Line 5 shows the actual contents of the quintiles performance classes we obtain with this random performance tableau instance. Notice the seed=5 argument in Line 2 which makes the random experiment repeatable.

As made evident with this example of 5-tiling, useful formal properties of our q-tiles sorting algorithm are the following:

1. *Coherence*: Each decision alternative is always sorted into a non-empty subset of adjacent q-tiles classes. For instance, alternative 'a16' is sorted into the best $(]0.8 - 1.0])$ and second best quintiles class $(]0.6 - 0.8])$. In the limit, a not yet evaluated alternative would appear sorted in all five quintiles classes.
2. *Uniqueness*: If no indeterminate outranking situation is being observed ($r() \neq 0$), a decision alternative is sorted into exactly one single quantiles class. This is the case, for instance, with alternative 'a24' which is solely sorted into the best quintiles class $(]0.8 - 1.0])$.
3. *Separability*: The quantiles class limits $\mathbf{q}(p^k)$ being given, the sorting result for each alternative x may be computed independently of the other alternatives' sorting results. Similarly, the content of a quantiles class $\mathbf{q^k}$ may be computed independently of the other classes' contents.

The last property will give us later on access to efficient *parallel processing* of class membership characteristics $r(x \in \mathbf{q^k})$ for all $x \in X$ and $k = 1, ..., k$.

1.3 A pre-ranked sparse approximation of the global outranking relation

Following the coherence property above, we may compute for each alternative x in X a *lower* and an *upper* q-tiles sorting limit. The lower limit is determined by the one of its lowest q-tiles class whereas the upper limit is determined by the one of its highest q-tiles class.

Reconsidering the quintiles sorting result of Example 2, we may observe, for instance, a decomposition of X into seven quantiles performance classes:

```
>>> pr.showDecomposition()
*--- quantiles decomposition in decreasing order---*
    c1. ]0.8-1.0]: [a24]
    c2. ]0.6-1.0]: [a16,a22, a32]
    c3. ]0.6-0.8]: [a02, a06, a25, a28, a31, a36, a37,
                    a39,  a40, a41, a48]
    c4. ]0.4-0.8]: [a01, a09, a10, a13, a18,
                    a27, a43, a45]
    c5. ]0.4-0.6]: [a03, a05, a07, a08, a14, a15, a20,
                    a26, a30, a33, a35, a38, a44, a49']
    c6. ]0.2-0.6]: [a04,'a11, a12, a17, a29, a34, a42,
                    a46, a47, a50]
    c7. ]0.2-0.4]: [a19, a21, a23]
```

Alternative 'a24', for instance, is sorted into the qantiles class $]0.8 - 1.0]$, whereas, alternative 'a16' is sorted into the quantiles class $]0.6 - 1.0]$.

The q-tiles sorting result leaves us hence with a partition of the set of decision alternatives into more or less refined quantiles performance classes. These classes may furthermore be linearly ranked from best to worst by following three potential ranking strategies:

1. *Optimistic*: In decreasing lexicographic order of, first, the upper and, secondly, the lower quantile;
2. *Pessimistic*: In decreasing lexicographic order of, first, the lower and, secondly, the upper quantile;

[5] $\vee$ denotes the symmetric or epistemic disjunction operator. If ϕ and φ denote two propositions, $r(\phi \vee \varphi)$ will be $\max(r(\phi), r(\varphi))$ if both $r(\phi)$ and $r(\varphi)$ are positive; $\min(r(\phi), r(\varphi))$ if both $r(\phi)$ and $r(\varphi)$ are negative; and 0 if $r(\phi)$ and $r(\varphi)$ are of opposite signs.

[6] Tutorials for programming with the DIGRAPH3 Python3 resources (see [10]) may be found on this site: http://leopold-loewenheim.uni.lu/docDigraph3/ .

3. *Average*: In decreasing lexicographical order of, first, the average of the lower and upper quantile, and secondly, the upper quantile.

Practical experiments with ranking given performance tableaux, like the one in Example 1 above, suggest that the '*average*' ranking strategy gives the most convincing result when indeterminate outranking situations and/or missing data are observed.

In view of the partition $\mathcal{P}_q = \{c_1, ..., c_k\}$ of the set X of decision alternatives, ranked from the best to the worst, we may define as follows the characteristic function of what we will call a *pre-ranked sparse* or *q-tiled* outranking relation, denoted $\gtrapprox^q$:

$$r(x \gtrapprox^q y) \stackrel{\text{def}}{=} \begin{cases} +1, & \text{if } [x \in c_i \wedge y \in c_j \wedge i < j] \\ -1, & \text{if } [x \in c_i \wedge y \in c_j \wedge i > j] \\ r(x \succsim_{|c_i} y) & \text{, otherwise.} \end{cases} \quad (6)$$

Relation $\succsim_{|c_i}$ denoting the restriction of the global outranking relation to the component c_i, $r(x \gtrapprox^q y) = r(x \succsim_{|c_i} y)$ only when x and y do appear in the same component c_i. The corresponding outranking digraph, denoted $G(X, \gtrapprox^q)$, is called a *pre-ranked* or *q-tiled* outranking digraph.

Depending on the number q of quantiles used in the q-tiles sorting algorithm, a more or less refined pre-ranking is obtained. In the limit, when, on the one hand, only one single component $c_1 = X$ is observed, we recover the standard outranking relation $\succsim$. On the other hand, when q is high and n singleton quantiles classes are obtained, we directly recover a linear ranking of the decision alternatives.

Reconsidering the performance tableau of Example 2, concerning a set of 50 decision alternatives, we show below a map of the standard outranking relation $\succsim$ (see Fig. 2. left side) versus a map of the corresponding pre-ranked outranking relation $\gtrapprox^5$ (see Fig. 2. right side).

Figure 2. Map of standard $\succsim$ outranking relation (left) versus corresponding pre-ranked, sparse approximate $\gtrapprox^5$ outranking relation (right) [8]

[8] Fig. 2. symbols legend: $\top \equiv$ outranking *for certain*; + $\equiv$ more or less *outranking*; ' ' $\equiv$ *indeterminate*; - $\equiv$ more or less *outranked*; $\bot \equiv$ outranked *for certain*.

The 5-tiled outranking relation of Example 2 shows indeed seven ordered components with a minimal cardinality of 1 (component c_1) and a maximal cardinality of 14 (component c_5). The outranking *fill rate* of the pre-ranked $\gtrapprox^5$ outranking relation, i.e. the actual remaining part in $\gtrapprox^5$ of the complete standard outranking relation $\succsim$ (see Fig. 2), amounts here to 18%. And the ordinal correlation index (see [7]) between the standard ($\succsim$) and the pre-ranked ($\gtrapprox^5$) outranking relation, denoted $\tau(\succsim, \gtrapprox^5)$ is, in this case, $+0.75$.

2 Computing linear rankings from a pre-ranked sparse outranking digraph

The previous quantiles sorting result represents in fact a first step in the construction of a global linear ranking of all given decision alternatives.

2.1 Heuristic linear closures of pairwise outranking situations

To linearly rank indeed the complete set X of decision alternatives, we still need to locally rank without ties the k components $c_i \subseteq X$ for $i = 1, ..., k$. To do so, we will rely on the component-wise restricted pairwise valued outranking situation characteristics, i.e. $r(x \succsim_{|c_i} y)$ for all x, y in c_i. We denote $\mathcal{G}_{|c_i}$ the restriction of the standard outranking digraph to the subset c_i of decision alternatives.

The component-wise ranking algorithm

1. **Input**: the outranking digraph $\mathcal{G}(X, \gtrapprox^q)$, a partition $P_q = \{c_1, c_2, ..., c_k\}$ of k linearly ordered decreasing parts of X obtained by the q-tiles sorting algorithm, and an empty list $\mathcal{R}$.
2. **For each** performance class $c_i \in P_q$:

   if $\#(c_i) > 1$:
   $R_k \leftarrow$ **locally rank** c_i in $\mathcal{G}_{|c_i}$ with Copeland's rule
   (if ties, render the concerned alternatives in alphabetic order)

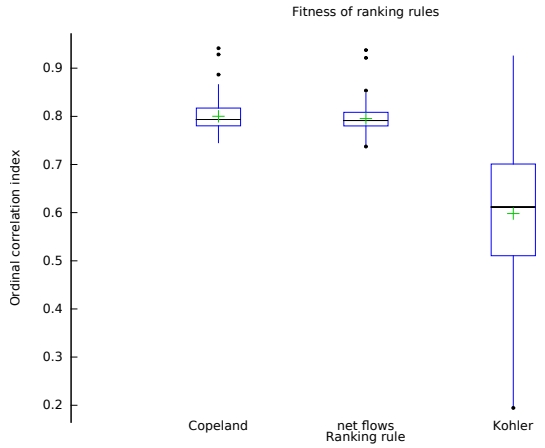
   else: $R_k \leftarrow c_i$

   append R_k to $\mathcal{R}$
3. **Output**: $\mathcal{R}$

The complexity of the component-wise ranking algorithm is *linear* in the number k of components resulting from a q-tiles sorting. Concerning the compexity of the local ranking procedure, we know that outranking relations do only exceptionally show linear rankings. Usually, they do not even render complete or, at least, transitive partial relations (see Bouyssou and Pirlot 2005 [11]). Three heuristic ranking rules appear most suitable here for our purpose –*Copeland*'s, *Net-flows*' and *Kohler*'s rule– all three of complexity $\mathcal{O}(\#(c_i)^2)$ on each restricted outranking digraph $\mathcal{G}_{|c_i}$. In case of *ties* (very similar evaluations for instance), the local ranking procedure will render these alternatives in increasing *alphabetic ordering* of their identity keys.

However, the three considered ranking rules do not deliver rankings of a same quality as we show in Fig. 3 below. We may notice, indeed, that the quality of the linear rankings obtained with Copeland's or the Net-Flows ranking rule on a sample of 100 outranking digraphs of order 250 is much better (median correlation with $\succsim$ around $+0.8$ [7]) when compared with Kohler's rule (median correlation with $\succsim$ around $+0.6$ only). As Copeland's ranking rule is the simplest to implement, we will by default use this ranking rule in the sequel.

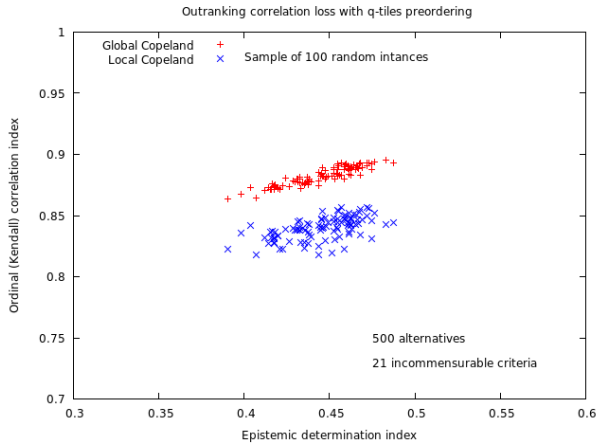
Figure 3. Sample of hundred 10-tiled outranking graphs of order 250



2.2 Fitness of the linear ranking result

The fitness of our pre-ranked sparse versus the standard outranking digraph model may be appreciated when comparing both the ordinal quality of the linear ranking result obtained from a standard versus the result obtained from a 50-tiled pre-ranked outranking relation. We consider a sample of 100 random performance tableaux gathering 500 decision alternatives evaluated on 21 incommensurable performance criteria. The 50-tiles pre-rankings give on average around 38 (sd. 7.6) performance classes, with a minimum of 32 and a maximum of 67 components, leading to an average diagonal outranking fill rate of around 3% (sd: 0.4%). Concerning the ordinal correlation (see [7]) between the standard outranking relation and our linear ranking result, we may notice in Fig. 4 that the correlation is around $+0.88$ (sd: 0.007) when the Copeland ranking is constructed from the global outranking relation, and around $+0.83$ (sd: 0.009), when it is constructed from the 50-tiled sparse outranking relation. The pre-ranking step, hence, does apparently not deteriorate significantly the quality of the ranking result.

Figure 4. Standard versus 50-tiled pre-ranked sparse outranking model



Notice furthermore in Fig. 4 that the ordinal correlations are, both times, augmenting with the epistemic determination index of the standard outranking relation (see [7]). It is indeed evident that the quality of any linear ranking result essentially depends on the actual performance evaluations. When, on the one hand, they are very similar or there are many missing data and/or vetoes, the asymmetric part of the outranking relation will appear weakly determined. All potential ranking rules will deliver rankings that are more or less *weakly* correlated with the corresponding outranking relation. On the other hand, if the evaluations show a clearly determined alignment of the decision alternatives, the outranking relation will appear strongly determined, and all potential ranking rules will produce more or less *highly* correlated linear rankings.

2.3 Multithreading the q-tiles sorting & ranking procedure

When tackling big performance tableaux, evaluating thousands or even millions of decision alternatives, we absolutely need to speed up the computations with multiple parallel threading HPC implementations. This is readily made possible by the *separability* property of the pairwise outranking approach.

Relying indeed on this property when sorting each alternative into its respective quantiles performance classes, the q-tiles sorting algorithm may be *safely split* into as much parallel processing threads as there are *parallel processing units* available. Notice that a high number of parallel processing units, sharing a same CPU memory, will consequently need a very big memory. Furthermore, the component-wise ranking procedures, being all local to a diagonal component c_i, may as well be safely processed in *parallel threads* on each restricted outranking digraph $\mathcal{G}_{|c_i}$.

Along these ideas, we have specially adapted our DIGRAPH3 computing resources[9] in order to run our q-tiles sorting & ranking algorithms on the HPC clusters of the University of Luxembourg (see [12]). The computations reported here were operated on the *gaia-80* node, a Delta D88x-M8-BI, 8 * Intel Xeon E7-8880 v2 @ 2.5 GHz machine equipped with 120 single threaded processing cores and a shared CPU memory of 3 TB.

The multithreaded versions of our algorithms are implemented with the help of the Python3.5 `multiprocessing` module[10] and our DIGRAPH3 collection of Python3 modules [10]. We use, for instance, the following generic algorithmic design for implementing parallel local ranking procedures.

Generic approach for parallel processing of the local rankings

```
from multiprocessing import Process, active_children
    class Thread(Process):
        def __init__(self, threadID, localComp)
                    Process.__init__(self)
                    self.threadID = threadID
                    self.localComponent
                    ...
        def run(self):
            ... Copeland ranking
            ... of self.localComponent
            ...

nbrOfJobs = number of // CPUs
for job in range(nbrOfJobs):
    ... pre-threading tasks per job
    print('iteration = ',job+1,end=" ")
    splitThread = Thread(job, localComponent, ...)
    splitThread.start()
while active_children() != []:
    pass
print('Exiting parallel threads')
for job in range(nbrOfJobs):
    ... post-threading tasks per job
```

In Table 1 we show run times of linear ranking constructions both, from a standard $\succsim$ outranking relation and, from a pre-ranked, sparse approximate $\gtrapprox^q$ outranking relation.

[9] See the documentation of the DIGRAPH3 resources FSTC/ILIAS Decision Systems Group, University of Luxembourg. http://leopold-loewenheim.uni.lu/docDigraph3 .

[10] See https://docs.python.org/2/library/multiprocessing.html .

Table 1. Comparing the standard and pre-ranked sparse approach[11]

digraph order	$\succsim$ relation run time	$\tau(>,\succsim)$	$\gtrapprox^q$ relation run time	$\tau(>^q,\succsim)$
1 000	6"	$+0.88$	4"	$+0.83$
2 000	15"	$+0.88$	9"	$+0.83$
2 500	27"	$+0.88$	14"	$+0.83$

If the speed gain for order 1 000 is 33%, we already reach nearly 50% acceleration with order 2 500. Notice that the quality, in terms of the correlation index with the gobal outranking relation, of the linear ranking result appears to be independent of the actual order, namely $+0.88$ for the standard, resp $+0.83$ for the pre-ranked outranking relation.

These excellent run times encouraged us to furthermore develop specific cythonized[12] C versions of our DIGRAPH3 Python modules in order to tackle pre-ranked outranking digraphs with sizes up to five trillions (5×10^{12}) of pairwise outranking situations (see Table 2). For the biggest instances, we generate a random 3 decision objectives performance tableau with 13 incommensurable criteria and 5% missing data.[13] For 1 000 000 alternatives (input data size = 1.4 GB) we thus obtain (see Table 2), with a 1 100-tiles pre-ranking, 13 547 diagonal components with a minimal size of 10 and a maximal size of 1 150 alternatives, leading to a fill rate of 0.05%. To linearly rank this huge digraph of size 10^{12} we need, on the gaia-80 machine with 120 parallel processing cores, in total about 1 hour and 17 minutes.

Table 2. HPC performance measurements for big data

$\gtrapprox^q$ outranking relation order	size	q	fill rate	run time
10 000	1×10^8	150	0.64%	13"
15 000	2.25×10^8	200	-	22"
25 000	6.25×10^8	300	-	39"
50 000	2.5×10^9	500	0.58%	2'
100 000	1×10^{10}	900	0.22%	5'
1 000 000	1×10^{12}	1100	0.05%	1h17'
1 732 051	3×10^{12}	1500	0.04%	3h09'
2 236 068	5×10^{12}	1600	0.03%	4h50'

Notice that the choice of the number q of quantiles is of crucial importance for our computations as it influences both the run times of the q-tiling as well as that of the local rankings. If q is relatively small, there will be less components, making on the one hand, the q-tiles sorting procedure quicker. Yet, some components might in consequence show much larger cardinalities, which make, on the other hand, these local ranking procedures quadratically slower. If q is chosen larger, q-tiling will get linearly slower. Yet, the local rankings might get much quicker in case there appear less components of larger cardinalities. Best overall run times are obtained in practice with a number q of quantiles that makes the q-tiling procedure take approximately the same run time as the local rankings.

Finally, we estimated the quality of such huge linear ranking, denoted $>_q$, by sampling the ranking quality on sub-digraphs of order 1 000 for instance. Here we recovered in fact, by the vertu of the LLN, an average sampled correlation result we have already noticed in Table 1 , namely $\overline{\tau}(\succsim,>_q) \rightsquigarrow +0.83$, with a standard deviation diminishing with the growth of the number of samples we take into account. This result confirms again that, given the same random performance tableau generator, the quality of our ranking does not depend on the actual order and size of the outranking digraph.

[11] Legend: $\tau(>,\succsim)$, resp. $\tau(>^q,\succsim)$ show the ordinal correlations between the corresponding linear ranking results, denoted $>$, resp. $<^q$, and the given standard outranking relation $\succsim$.

[12] Cython: C-extensions for Python: http://cython.org/ .

[13] On generating random performance tableaux. See Tutorials of the DIGRAPH3 ressources .

Conclusion

We present in this paper a sparse, approximate outranking digraph model coupled with a two steps ranking algorithm based on quantiles-sorting & local-ranking procedures. Ranking results obtained with this outranking digraph model fit well with those given by a standard outranking digraph. Furthermore, separable quantiles sorting and local ranking procedures offer effective multiprocessing capacities. Efficient *scalability* allows, hence, the linearly ranking of very big performance tableaux gathering up to millions of evaluations. Good perspectives for further optimization with cython C implementations and HPC ad hoc tuning are given. All Python and cython HPC modules are freely available for further developments on the git repository of the DIGRAPH3 resources: http://github.com/rbisdorff/Digraph3 .

Acknowledgments. The author would like to thank the UL-HPC administration team for their helpful and competent assistance in the fine tuning of our HPC work. The anonymous reviewers, with their pertinent comments and suggestions, also much helped enhancing content and readability of this paper.

REFERENCES

[1] N. Japkowicz and J. Stefanowski (2015). *Big Data Analysis: New Algorithms for a New Society*. Springer-Verlag Berlin Heidelberg, Studies in Big Data, 329 pages.

[2] R. Bisdorff, L.C. Dias, P. Meyer, V. Mousseau and M. Pirlot (Eds.) (2015). *Evaluation and decision models with multiple criteria: Case studies*. Springer-Verlag Berlin Heidelberg, International Handbooks on Information Systems, DOI 10.1007/978-3-662-46816-6-1, 643 pages.

[3] B. Roy (1991). The outranking approach and the foundations of the Electre methods. *Theory and Decision*, 31(1):49–73.

[4] R. Bisdorff (2000). Logical foundation of fuzzy preferential systems with application to the electre decision aid methods. *Computers and Operations Research* (Elsevier) 27:673–687.

[5] R. Bisdorff (2002). Logical Foundation of Multicriteria Preference Aggregation. In: Bouyssou D et al (eds) *Aiding Decisions with Multiple Criteria*. Kluwer Academic Publishers, pp 379–403.

[6] S. S. Wagle, M. Guzek, P. Bouvry and R. Bisdorff (2015). An Evaluation Model for Selecting Cloud Services from Commercially Available Cloud Providers. In Proceedings of the 7th IEEE *International Conference on Cloud Computing Technology and Science*, Vancouver, Canada, November 30 - December 2 2015, ISBN 978-1-4673-9560-1/15, pp 107–114.

[7] R. Bisdorff (2012). On measuring and testing the ordinal correlation between bipolar outranking relations. In Proceedings of DA2PL'2012 - *From Multiple Criteria Decision Aid to Preference Learning*. University of Mons, November 15-16, pp 91–100.

[8] B. Roy and D. Bouyssou (1993). *Aide Multicritère à la Décision : Méthodes et Cas*. Economica Paris, 695 pages.

[9] R. Bisdorff (2013) On polarizing outranking relations with large performance differences. *Journal of Multi-Criteria Decision Analysis* (Wiley) 20:3-12.

[10] R. Bisdorff (2016). Tutorials of the DIGRAPH3 resources. *Documentation of the DIGRAPH3 resources* FSTC/ILIAS Decision Systems Group, University of Luxembourg. http://leopold-loewenheim.uni.lu/docDigraph3 .

[11] D. Bouyssou and M. Pirlot (2005). A characterization of concordance relations. *European Journal of Operational Research* 167: 427–443.

[12] S. Varrette, P. Bouvry, H. Cartiaux and F. Georgatos (2014). Management of an Academic HPC Cluster: The UL Experience. In Proc. of the 2014 Intl. Conf. on *High Performance Computing & Simulation* (HPCS 2014), Bologna (Italy). IEEE, pp 959–967.

Metrics and Experiment Organization of Multiuser Preference Learning in Recommender Systems and Decision Aid

Michal Kopecky [1] and **Ladislav Peska** [2] and **Peter Vojtas** [3] and **Marta Vomlelova** [4]

1 Extended Abstract

We consider user-item preference represented by a rating (ordered by rating/score). That is, for a set of users U and items I the preference is a rating function

$$r : U \times I \longrightarrow [0, 1].$$

In case of observed ratings r^o, this function is partial and the range size is usually ≤ 10 (e.g. 10 stars rating with a lot of ties). In case of recommendation (estimation, generalization, preference learning) r^e this function is expected to be total and usually produces linear ordering. Success of this inductive process is measured on concordance between r^o and r^e (with classical division to train, test and/or validation sets and cross-validation). This concordance can be measured by several metrics and measures. Learning specifically preferences of a single user $u \in U$ we consider evaluation restricted to

$$r(u, .) : I \longrightarrow [0, 1].$$

We focus on content based recommendation and multiple user scenario. Content based recommendation assumes that items have attributes $A_1, \ldots, A_m$ with domain $dom(A_i) = D_i$ and a record looks like $(id, a_1, \ldots, a_m)$, where $a_i = id.A_i$. We model preference of a user $u \in U$ by $u^{t^u f^u}$ by a monotone combination t^u of local utilities $f_i^u : D_i \longrightarrow [0, 1]$. So estimated user u's preference on item id is given by

$$r(u^{t^u f^u}, id) = t^u(f_1^u(id.A_i), \ldots, f_m^u(id.A_m))$$

In [3] we have shown that this representation is unique for triangular f_i^u and arbitrary t^u. Complementary, in [4] uniqueness is shown for preferences where t^u is restricted to Choquet integral and f_i^u is (almost) arbitrary.

In terminology of [2], this is a prediction of ordering of a fixed set of elements. Hence can be considered also as a label ranking problem (instances are users and labels are items, ties are broken arbitrary, we adopt this to position error). Nevertheless some similarities with instance ranking and object ranking can be found too.

In [10, 11] we use movie recommendation data (integrated MovieLens, Netflix, RecSys2014 competition and IMDB datasets). Here the user's visit return frequency is much higher than in experiments in [8]. In [8] we use real world production data (with implicit user behavior collected by scripts interpreted as a (fictitious) rating) made public by L. Peska in [6, 7] (so experiments are repeatable).

Distinguishing feature is that besides classical ranking loss metrics we consider also new position error metrics (in terminology of [2]). For instance (parallel) first hit is the step in which a top-k item (from test set) appears in top-k of our estimation (the smaller the number the better).

We measure quality of our prediction also for each user separately. This can be represented by diagrams depicting quantiles on users with certain first hit recommendation bellow some value (see e.g. Figure 1).

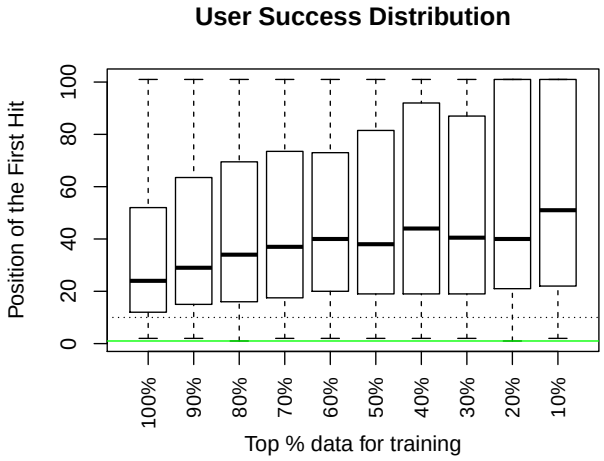


Figure 1. Distribution of users according First hit measure for different models (learned on a data sub-sample).

We see that there are users easier to recommend (e.g. in first quartile) but also users more complicated (e.g. in last quartile). Nevertheless, this does not generate a linear order of methods, e.g. one method can be better at medium quantile and worse at first quartile than other.

Considering portion of users which display also further pages of recommendation results, we can mimic global performance of our method over the whole user-item matrix. Denote p_j^m % of users with 1^{st} hit$^m \in (10 * (j-1), 10 * j]$ and q_j % of users visiting j^{th} page (probably dependent on domain). We introduce a new Aggregated Successful Visit measure as $p_1 * q_1 + p_2 * q_2 + p_3 * q_3 \ldots$ (see Figure 2).

[1] Charles University, email: kopecky@ksi.mff.cuni.cz
[2] Charles University, email: peska@ksi.mff.cuni.cz
[3] Charles University, email: vojtas@ksi.mff.cuni.cz
[4] Charles University, email: marta@ktiml.mff.cuni.cz

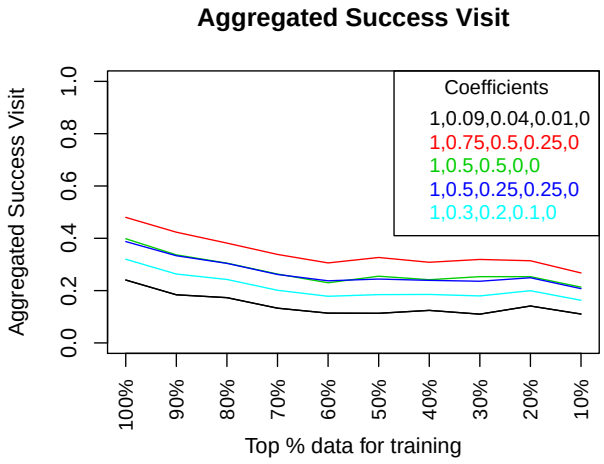


Figure 2. Aggregated success visit for different domains, characterized by portions of users visiting i^{th} page and method results.

Most of experiments are off-line experiments and/or simulations on artificial data. In [8] we report also on some online A/B tests.

Our methods are combination of heuristics and variations of data mining methods. Methods are tuned with respect to final evaluation metric. For order sensitive (e.g. position error) metrics these gives new alternative methods of learning.

In [9] authors state: Several methods have been proposed in the past decades to deal with Multicriteria Decision Aiding (MCDA) problems. Even if the axiomatic foundations of these methods are generally well known, comparing the different methods or simply analysing the results produced by the methods on real-life problems is arduous as there is a lack of benchmark MCDA datasets in the literature. In [1] authors suppose that the preference information provided by the decision maker is indirect and has the form of pairwise comparisons of criteria with respect to their importance and pairwise preference of alternatives (a new axiomatic foundation). They propose a simulation, where decision maker's answers depend and are coherent on/with training (artificial) data. This gives a new possibility of verification and mutual comparison of multiple criteria decision aid methods. Can our real world data from [6, 7] be used in experiments? Main challenge is experiment organization. Namely, it is not clear, when human answers are in concordance with data. Maybe experiences collected in [5], where human operator of furnace linguistic answers where used, can be helpful.

We conclude, that variations on metrics and experiment organization can make preference learning closer to real world recommendation needs.

Acknowledgements. We announce partial support of Czech grant P46.

REFERENCES

[1] S. Angilella, S. Corrente, S. Greco, and R. Słowiński, *Robust ordinal regression and stochastic multiobjective acceptability analysis in multiple criteria hierarchy process for the Choquet integral preference model*, *Omega*, **63**, 154 – 169, 2016.

[2] J. Fürnkranz and E. Hüllermeier, *Preference Learning: An Introduction*, 1–17, Springer Berlin Heidelberg, Berlin, Heidelberg, 2011.

[3] M. Kopecky, M. Vomlelova, P. Vojtas. *Basis Functions as Pivots in Space of Users Preferences*, 45–53, Springer, In New Trends in Databases and Information Systems, Volume 637 of the series Communications in Computer and Information Science, 2016.

[4] Ch. Labreuche, E. Hullermeier, P. Vojtas, and A. Fallah Tehrani, *On the identiffability of models in multi-criteria preference learning*, this volume, 2016.

[5] V. Novak and J. Kovář, *Linguistic If-Then Rules in Large Scale Application of Fuzzy Control*, In Fuzzy If-Then Rules in Computational Intelligence, Theory and Applications, Da Ruan, E. E. Kerre (Eds.), 223–241, Springer International Series in Engineering and Computer Science, 553, 2000.

[6] L. Peska. Secondhand bookshop dataset. http://www.ksi.mff.cuni.cz/ peska/bookshop2015.zip

[7] L. Peska. Travel agency dataset http://www.ksi.mff.cuni.cz/ peska/travelagency2015.zip , 2015.

[8] L. Peska and P. Vojtas, *Using implicit preference relations to improve recommender systems*, *Journal on Data Semantics*, 1–16, (2016).

[9] A. Rolland and J. Cugliari, *Generating new multicriteria data*, this volume, (2016).

[10] P. Vojtas, M. Kopecky, and M. Vomlelova, *Understanding Transparent and Complicated Users as Instances of Preference Learning for Recommender Systems LNCS 9548*, 23–34, Springer International Publishing, 2016.

[11] P. Vojtas, M. Kopecky, and M. Vomlelova. Understanding transparent and complicated users in content based recommendation, EURO 2016 Poznan, Preference learning stream, 2016.

Interpretable Score Learning by Fused Lasso and Integer Linear Programming

N. Sokolovska[1], **Y. Chevaleyre**[2], **J.-D. Zucker**[3]

Abstract. Score learning aims at taking advantage of supervised learning to estimate interpretable models which facilitate decision making. Ideally, a scoring system is based on simple arithmetic operations, is sparse, and can be easily explained by human experts.

In this contribution we introduce an original methodology to simultaneously learn interpretable binning mapped to a class variable, and the scores associated with these bins. We show by numerical experiments on standard data sets that our approach is competitive compared to the state-of-the-art methods.

1 Introduction

Traditionally, a problem in supervised machine learning is presented as a binary or multi-class classification where the goal is to learn real valued weights of a model. However, although the generalizing error is an important criterion, in some applications the interpretability of a model plays even a more significant role. Among such tasks are e.g. medical diagnosis, biomedical challenges coming from quantitative metagenomics, where data are represented as abundance matrices of bacteria, and other tasks challenged by medical community.

Since recently, machine learning is actively used in decision making, and problems on their intersection are an active field of research [16]. We are in particularly interested to learn scoring systems which are defined as linear classification models that do prediction using basic arithmetical operations such as addition, subtraction, and multiplication, and where only a limited number of variables is needed [17]. Learning automated medical scores is rather new domain of research, and the extensive literature does not really exist. Among the state-of-the-art methods are Supersparse Linear Integer Models (SLIM) developed by [17] for automated medical score learning, which are reported to perform efficiently on various data sets. The model is formulated as an integer programming task and optimizes directly the accuracy, the 0-1 loss, and the degree of sparsity. Among interesting but not very related papers, since we do not dispose of time series, we would like to mention [3] where risk scores (disease severity score learning, or DSSL) are learned from time series data to model progression of disease. Another interesting avenue of research is Bayesian-based approaches to learn scoring systems. So, [4] introduced a Bayesian model where a prior favours fewer significant digits, and therefore the solution is sparse. A Bayesian model is also developed in [18] to construct a falling rule list, which is a list of simple if-then rules containing a decision-making process and which stratifies patients from the highest at-risk group to the lowest at-risk group. A similar idea based on Bayesian learning is considered by [8] where the main motivation is to construct simple rules which are interpretable by human experts and can be used by healthcare providers.

The state-of-the-art methods of [4] and [17] are reported to be efficient, but an obvious drawback is that their output, the learned scores, apply to real-valued data (if the input data were real). Although medical data are often indeed real, a model which provides some interpretable discretization or learns *diagnostic thresholds*, is of a bigger interest for diagnostic purposes. A mixed integer program can help to assign objects to categories [14].

The difficulty to learn discrete classifiers is well described by [1], and comes from the fact that the problem is equivalent to the linear integer programming problem which is NP-complete. Real-valued standard classifiers on the contrary are known to be cast as convex optimization problems and are able to find a solution in a polynomial time. [1] propose a scheme to learn binary-weighted linear functions with a hinge loss which is based on randomized rounding to find fractional solutions. Although randomized rounding is not a recent technique (see [13]), it is a natural way to find discrete solutions efficiently. So, [5] also use the randomization as a core idea for a large-scale learning. They [5] made a first attempt to implement online learning with ranomized rounding, where the main motivation is to reduce the memory needed to store the vector of coefficients of a parametric model, since storing all the coefficients consumes considerable RAM.

In our contribution we propose an efficient method which performs a supervised discretization and the score learning for the optimal bins simultaneuously. The fused lasso penalty term is an important element of our approach. *The fused lasso shrinks similar variables to each other creating bins, and ordering them.*

This paper is organised as follows. In Section 2 we discuss ways to perform supervised categorization, and we introduce our method. We provide also the linear programming formulation for it. Section 3 discusses some state-of-the-art related methods which provide us with the baseline comparative performance. We show the results of the numerical experiments in Section 4. Concluding remarks and perspectives close the paper.

2 Score Learning and Discretization by Fused Lasso

In this section we discuss how to learn scores associated with groups in categorical data (Section 2.1), and we introduce our approach which is formulated as an integer linear programming problem where we minimize the hinge loss penalized by the fused lasso penalty term (Sections 2.2 and 2.3).

[1] University Paris 6, INSERM, NutriOmics, Paris, France
[2] University Paris 13, LIPN, Villetaneuse, France
[3] IRD, NutriOmics, Bondy/Paris, France

2.1 Score Learning from Fully Categorical Data

Probably the most direct way to obtain an interpretable model is to deal with data which are already discrete or discretized. In this case it is sufficient to learn scores associated with the categories. In real applications data are indeed sometimes categorical, but quite often we deal with continuous or mixed data where a part of variables is categorical and another part is continuous. A natural way is to discretize a data set first and to work further with the discretized data. Such an approach which includes two steps – 1) Discretize data if necessary, and 2) Learn weights for each category – is simple, robust but may be suboptimal since it obviously returns an approximate solution.

An efficient way to perform supervised discretization is e.g. the recursive partitioning [2, 6, 7] which is based on a tree-structured regression. Its efficiency comes from its statistical properties letting not to evaluate all $2^{K-1}-1$ possible splits of a covariate at K levels [6]. In general, the recursive partitioning builds a decision tree whose goal is to classify as correctly as possible observations by splitting them into sub-populations. The algorithm is called recursive since the sub-populations are in their turn also split potentially infinite number of times.

The second step – score optimization from the categorical data – can be efficiently done by means of linear integer programming where we aim to

$$\text{minimize } \frac{1}{2}x^tQx + c^tx, \tag{1}$$

$$\text{such that } Ax \leq b, \tag{2}$$

$$lb \leq x \leq ub. \tag{3}$$

If the following and in our experiments we suppose that $Q = \mathbf{0}$, and we solve a linear problem. In the experiments, we use the IBM ILOG CPLEX Optimization Studio[4] which performs the constrained optimization.

2.2 Supervised Discretisation Solved by Integer Linear Programming

In our setting, the training procedure has access to N i.i.d. labeled observations $(X_i, Y_i)_{i=1}^N$. We assume that the class variables Y take their values in a finite set $\mathcal{Y}$. Here, without loss of generality, we consider the binary classification problem, and $\mathcal{Y} = \{0, 1\}$. We also assume that the observations X take their values in $\mathcal{X} \in \mathbb{R}^d$.

A variable to discretise is a column from the observation matrix (from a potentially very big data matrix). Here, X is a vector of size $N \times 1$. The Algorithm 1 sketches the procedure which we propose to use to perform the supervised discretization.

Algorithm 1 Supervised Categorization Solved by Integer Linear Programming

Input: X, Y
Output: weights associated with each (observed) value in X
V contains all unique values of X
Initialise matrix $A = \mathbf{0}$ (dimension $N \times |V|$)
for each $V_j, j \in \{1, \ldots, |V|\}$ **do**
  $A_{(\mathbb{1}_{X==V_j}, j)} = 1$
end for
Use matrix A as a covariate matrix in an optimisation procedure (e.g. hinge loss minimisation) to find $f : A \to Y$

[4] http://www-03.ibm.com/software

The linear programming formulation takes the following form. We minimise the hinge loss

$$\sum_{i=1}^{N} \boldsymbol{\ell}(y_i, \theta \cdot x_i + b), \tag{4}$$

we have $N + |V| + 1$ variables $\theta_1, \ldots, \theta_{|V|}, b, \xi_1, \ldots, \xi_N$ in this task, and the optimisation problem can be written as follows

$$\min \sum_{i=1}^{N} \xi_i, \text{ such that} \tag{5}$$

$$\text{for all } i,\ y_i(\theta \cdot x_i + b) \geq 1 - \xi_i, \tag{6}$$

$$\xi_i \geq 0. \tag{7}$$

2.3 Fused Lasso in Score Learning

The fused lasso penalty was introduced by [12] as a generalization of lasso or L_1 penalty term

$$\lambda \sum_{j=1}^{d-1} |\theta_j - \theta_{j+1}|, \tag{8}$$

where λ is a hyper-parameter controlling sparsity. The idea behind the fused lasso was to order features with the help of the regularization. The fused lasso penalizes the L_1 norm of both the coefficients and their successive differences. As a result, the penalty shrinks neighbouring coefficients towards each other. Hence, *similar variables are grouped or binned, having the same optimized weight*.

Here, Algorithm 1 is still applied to the continuous data, however, the linear programming formulation with the fused lasso term takes the following form. We minimise the hinge loss

$$\sum_{i=1}^{N} \boldsymbol{\ell}(y_i, \theta \cdot x_i + b) + \lambda \sum_{j=1}^{d-1} |\theta_j - \theta_{j+1}|, \tag{9}$$

and we get $|V| + 1 + N + (d-1)$ variables $\theta_1, \ldots, \theta_{|V|}, b, \xi_1, \ldots, \xi_N, \eta_1, \ldots, \eta_{d-1}$. If we re-write the task as an optimisation problem, we obtain:

$$\min \big(\sum_{i=1}^{N} \xi_i + \sum_{j=1}^{d-1} \eta_j \big), \text{ such that} \tag{10}$$

$$\text{for all } i,\ y_i(\theta \cdot x_i + b) \geq 1 - \xi_i, \tag{11}$$

$$\text{for all } j,\ -\lambda\eta_j \leq \theta_j - \theta_{j+1} \leq \lambda\eta_j, \tag{12}$$

$$\xi_i \geq 0. \tag{13}$$

3 Some Related Methods

In this section we provide some details on two state-of-the-art approaches to which we compare our results, namely the online gradient descent with randomized rounding and SLIM.

Algorithm 2 sketches the online gradient descent with randomized rounding proposed by [5].

The SLIM introduced by [17] minimizes directly the $0-1$ loss and the L_0 penalty:

$$\min_{\theta} \frac{1}{N} \sum_{i=1}^{N} \mathbb{1}\{y_i\theta^t x_i \leq 0\} + \lambda_0\|\theta\|_0 + \lambda_1\|\theta\|_1, \tag{14}$$

Algorithm 2 Online Gradient Descent with Randomized Rounding [5]

Initialize $\theta = \mathbf{0}$, set constant R
for $t = 1 : T$ // for T iterations **do**
  Compute the gradient ∇_{θ_t}
  θ_{t+1} = Project($\theta_t - \alpha_t \nabla_{\theta_t}$)
  $\hat{\theta}_{t+1}$ = RandomizedRounding(θ_{t+1},ϵ_t)
end for
function Project(θ) = $\max(-R, \min(\theta, R))$
function RandomizedRounding(θ,ϵ)
  $b = \epsilon \left\lceil \frac{\theta}{\epsilon} \right\rceil$
  $a = \epsilon \left\lfloor \frac{\theta}{\epsilon} \right\rfloor$
  return b with probability $(\theta - a)/\epsilon$,
  and a with probability $1 - (\theta - a)/\epsilon$

where λ_1 is supposed to be chosen very small, since its role is only to restrict coefficients to coprime numbers. The task is presented and solved as an integer programming problem, we use the MatLab implementation[5] provided by the SLIM authors.

4 Experiments

In this section we illustrate the results of the above described approaches on a tiny synthetic problem, and on some standard data sets.

4.1 A Toy Example

We model a simple situation where there are two classes $\{-1, +1\}$, the number of observations is small, and the number of possible (observed) values is even smaller. There are 10 observations, and four unique (continuous) values of X: $0.1, 0.25, 0.63, 0.9$. The classes are well-separated, since the observations $<= 0.5$ belong to class -1, and those with > 0.5 belong to class $+1$.

Let

$$X = \begin{bmatrix} 0.1 \\ 0.1 \\ 0.1 \\ 0.25 \\ 0.25 \\ 0.63 \\ 0.63 \\ 0.9 \\ 0.9 \\ 0.9 \end{bmatrix} \qquad Y = \begin{bmatrix} -1 \\ -1 \\ -1 \\ -1 \\ -1 \\ +1 \\ +1 \\ +1 \\ +1 \\ +1 \end{bmatrix} \tag{15}$$

If we enumerate all observed values and construct matrix A which will help us to learn the weights (Algorithm 1), we get:

$$\text{A} = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \\ 7 \\ 8 \\ 9 \\ 10 \end{array} \begin{array}{c} \begin{array}{cccc} = 0.1 & = 0.25 & = 0.63 & = 0.9 \end{array} \\ \begin{pmatrix} 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \end{pmatrix} \end{array}$$

[5] https://github.com/ustunb/slim-matlab

We find solutions using SLIM, hinge loss gradient descent, hinge loss gradient descent with randomized rounding, unpenalized hinge loss solved with linear programming ("Rcplex" R package), and the fused lasso hinge loss solved by linear programming. The results of the optimization are as follows:

- SLIM: $(-9, -10, -10, 10, 10)$ (the first value corresponds to the intercept)
- with hinge loss (no regularization), the fused lasso, and with the hinge loss gradient descent with randomized rounding (100 repetitions, the average): $(-1, -1, 1, 1)$
- with hinge loss gradient descent $(-1.02, -1.00, 1.02, 1.00)$

It is easy to see that all methods cope with the task very well, since the problem is very simple. The hinge loss solved by the linear programming and the one penalized by the fused lasso find the same solution, what is not surprising in this case. The randomized rounding gradient descent converges to the same optimum. The continuous gradient descent converges to a continuos solution what is not very practical for scoring systems. We observe that solution of SLIM lies in a much bigger interval, what is unnecessary and can be unpractical in real applications.

4.2 Results on Benchmark Data

In this section, we share our results on some standard data sets, all downloadable from the UCI Machine Learning repository[6] [9].

- Mushroom Data Set is completely discrete, the number of instances is 8124, and the number of attributes is 22. The goal is to predict whether mushrooms are poisonous or edible.
- Qualitative Bankruptcy. The data are categorical, with 250 observations and 7 attributes such as industrial risk, management risk, competitiveness, etc.
- Breast Cancer Wisconsin (Prognostic) Data Set where we dispose of about 30 parameters describing characteristics of the cell nuclei present in the medical images for 198 patients [15]. All parameters are continuous.
- Haberman's Survival Data Set where the aim is to classify patients according to the survival outcome after surgery for breast cancer. There are only three attributes (age of patient, year of operation, and the number of positive auxiliary nodes detected) for 306 patients.
- Mammographic Mass Data Set is dedicated to discrimination of benign and malignant mammographic masses [10]. Among 5 attributes only 1 is continuous, and the other are categorical. The number of patients is 961.
- Spambase Data set contains 4601 emails which can be divided into two categories (spam or not spam). All 57 attributes are either continuous (normalized word frequencies) or integers (number of capital letters, longest sequence of capital letters, etc.)
- Ionosphere Data set contains 351 observations and 34 attributes to discriminate good and bad ionosphere conditions. The observations are radar data, and they are continuous.
- Glaucoma diagnosis set includes data from laser scanning images taken from the eye background for 170 patients (66 attributes). The data is part of the "ipred" R package [11].

We perform 10-fold cross validation and plot accuracy on training and test data. Figure 1 shows accuracy of the approach where we discretize the data if they are not categorical by the recursive partitioning, and then we minimize the hinge loss using linear programming.

[6] http://archive.ics.uci.edu/ml/

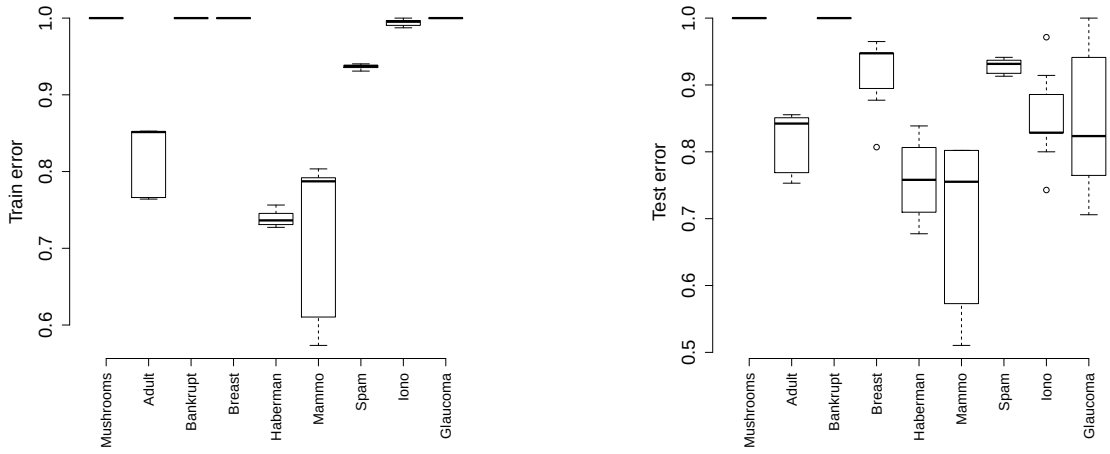


Figure 1. Performance on the UCI repository data. Data are discretized columnwise by Recursive Partitioning ("smbinning" R package) if necessary. The hinge loss minimization is applied to the categorical data and is done by the linear programming. On the left: training error, on the right: testing error. On some data sets such as Mushrooms and Bankruptcy the method achieves the state-of-the-art performance. On some data sets, on the contrary, we observe overfitting.

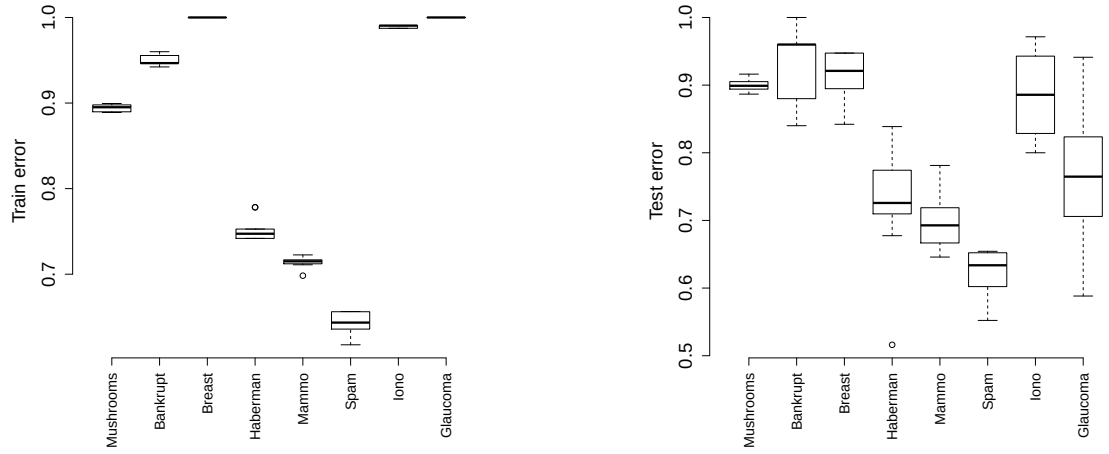


Figure 2. Performance on the UCI repository data. Results of the proposed supervised discretization optimized by the linear programming (CPLEX optimisation). The columns containing continuous variables are discretized using Algorithm 1 and the linear formulation presented in Section 2.2 is applied. On the left: training error, on the right: testing error.

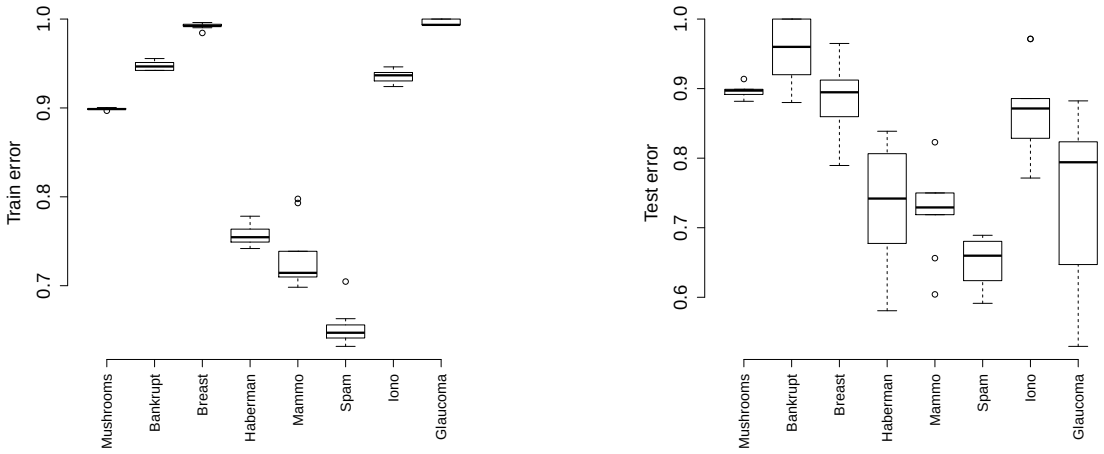


Figure 3. Results of the supervised discretization based on the linear programming penalized by the fused lasso. The linear formulation of the method is described in Section 2.3. On the left: training data, on the right: testing data.

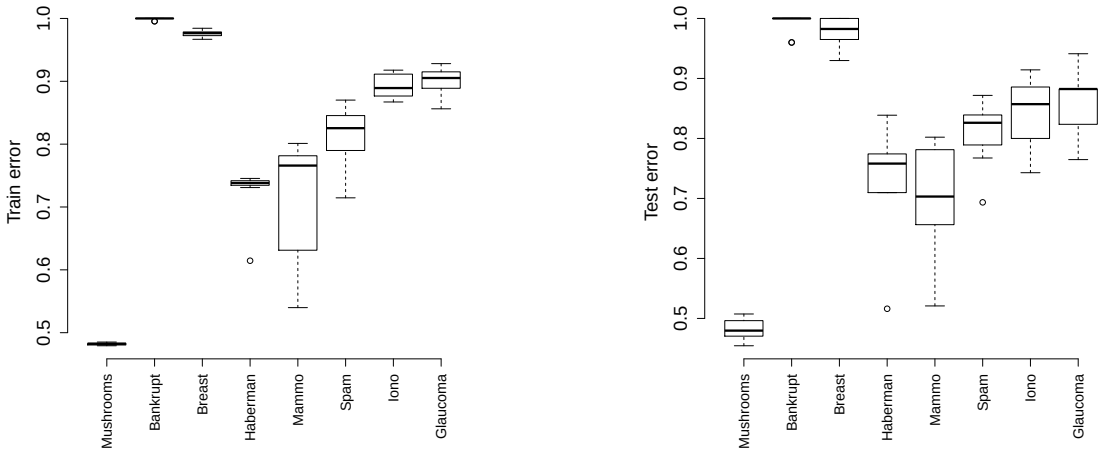


Figure 4. Results of the online gradient descent with the randomized rounding. The procedure used is presented as Algorithm 2. On the left: training data, on the right: testing data.

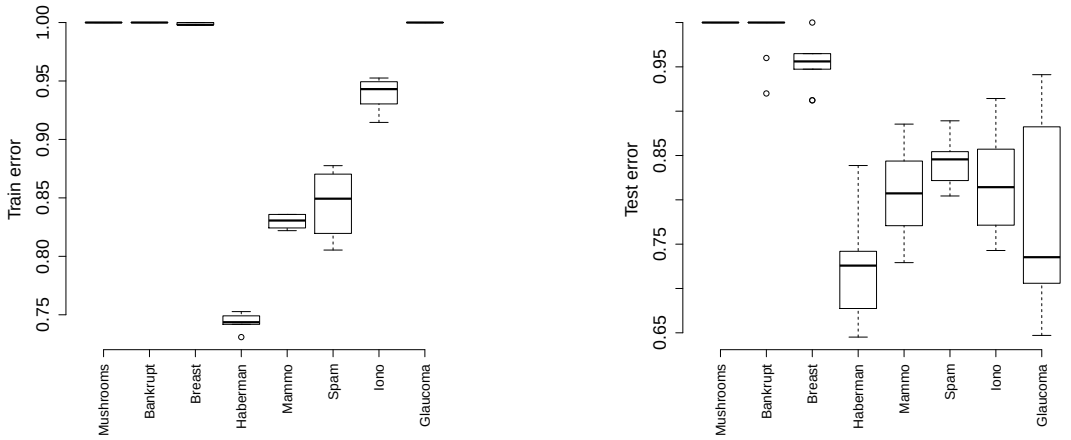


Figure 5. Results of SLIM. On the left: training data, on the right: testing data.

The recursive partitioning is done with the "smbinning" R package[7] which is developed to categorize numeric variables into bins with respect to binary class variables.

Figure 2 illustrates the results of the proposed supervised discretization. The hinge loss function is used but no penalty is applied. Note that although the performance is quite reasonable, the weights learned by this method are not practical, since the solution is not sparse and the values are not monotonous. Figure 3 plots the accuracy of the version which is penalized by the fused lasso. In this case, on the contrary, the solution is sparse and the variables are naturally put into bins, having same weights after the optimization procedure.

Figures 4 and 5 demonstrate the performance of the state-of-the-art methods. The performance of the online gradient descent with the randomized rounding is shown on Figure 4. The optimisation procedure is drafted as Algorithm 2, and corresponds to the method proposed by [5]. The SLIM is an efficient system to learn scores, however, if the hyperparameters are not well chosen, the approach suffers from sever overfitting. The results of the SLIM experiments are plotted on Figure 5.

5 Conclusion

In this contribution, we have tried to address the problem of efficient score learning from continuous data where we simultaneously learn reasonable thresholds to bin data and to learn the corresponding scores. Our main result is presented in Section 2.3, where we propose to use integer linear programming to optimize the hinge loss function penalized by the fused lasso. We have shown by experiments on some standard data from the UCI machine learning repository that the novel method is promising and competitive compared to several state-of-the-art methods. However, if the number of observed (continuous) values is very high in data, the optimization can become very complex or even not tractable by the integer linear programming. Another open issue is the detailed theoretical analysis of the proposed approach.

[7] https://cran.r-project.org/web/packages/smbinning/index.html

REFERENCES

[1] Y. Chevaleyre, F. Koriche, and J.-D. Zucker. Rounding methods for discrete linear classification. In *ICML*, 2013.

[2] E.F. Cook and L. Goldman. Empiric comparison of multivariate analytic techniques: advantages and disadvantages of recursive partitioning analysis. *Journal of chronic diseases*, 37(9–10), 1984.

[3] K. Dyagilev and S. Saria. Learning (predictive) risk scores in the presence of censoring due to interventions. *Machine Learning*, 2016.

[4] S. Ertekin and C. Rudin. A Bayesian approach to learning scoring systems. *Big Data*, 3(4), 2015.

[5] D. Golovin, D. Sculley, H. B. McMahan, and M. Young. Large-scale learning with less ram via randomization. In *ICML*, 2013.

[6] T. Hothorn, K. Hornik, and A. Zeileis. Unbiased recursive partitioning: a conditional inference framework. *Journal of Computational and Graphical Statistics*, 15(3), 2006.

[7] K.E. James and H.C. Kraemer. Repeated split sample validation to assess logistic regression and recursive partitioning: an application to the prediction of cognitive impairment. *Statistics in medicine*, 24(19), 2006.

[8] B. Letham, C. Rudin, T. McCormick, and D. Madigan. Building interpretable classifiers with rules using Bayesian analysis. *Annals of applied statistics*, 2015.

[9] M. Lichman. UCI machine learning repository, 2013.

[10] R. Schulz-Wendtland M. Elter and T. Wittenberg. The prediction of breast cancer biopsy outcomes using two cad approaches that both emphasize an intelligible decision process. *Medical Physics*, 34(11), 2007.

[11] A. Peters, T. Hothorn, and B. Lausen. ipred: Improved predictors. *R News*, 2(2), 2002.

[12] Tibshirani R, M.Saunders, S. Rosset, J. Zhu, and K. Knight. Sparsity and smoothness via the fused lasso. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 67(1), 2004.

[13] P. Raghavan and C.D. Tompson. Randomized rounding: a technique for provably good algorithms and algorithmic proofs. *Combinatorica*, 7(4), 1987.

[14] O. Sobrie, V. Mousseau, and M. Pirlot. Learning the parameters of a non compensatory sorting model. *Algorithmic Decision Theory*, 9346:153–170, 2015.

[15] W. N. Street, O. L. Mangasarian, and W.H. Wolberg. An inductive learning approach to prognostic prediction. In *ICML*, 1995.

[16] T. Tulabandhula and C. Rudin. On combining machine learning with decision making. *Machine learning*, 2014.

[17] B. Ustun and C. Rudin. Supersparse linear integer models for optimized medical scoring systems. *Machine Learning*, 2015.

[18] F. Wang and C. Rudin. Falling rule lists. In *AISTATS*, 2015.

Socially Conscious Consumption: Consumers' Willingness-To-Pay for Fair Trade Labels

Friederike Paetz[1] and **Daniel Guhl**[2]

Abstract. The consumption of Fair Trade (FT) products has increased tremendously in recent years. This implies a shift in preferences for FT products. But, how high is the consumers' willingness-to-pay (WTP) for the FT attribute and do differences in WTP emerge from consumers' demographic or psychographic variables? To answer these research questions concerning preferences for FT products and the underlying sources of preference shifts, we focus on the German orange juice market and determine consumers' WTP for a FT label by conducting a discrete choice experiment. We estimate mixed logit models (using regression-based approaches) and highlight the importance of accounting for observed and unobserved consumer heterogeneity in utility functions. We found an average WTP for a FT label of 23 Eurocent (17.4 % price premium), which markedly varies with consumer's age and his/her consciousness for fair consumption.

1 Introduction

Socially conscious consumption, e. g., the consumption of Fair Trade (FT) products has increased enormously in the last decade (Tully & Winer, 2014). In comparison to traditionally traded products, FT products adhere the guidelines of the FLO (Fairtrade Labelling Organization). Those guidelines inter alia safeguard the rights of workers and producers, and fix labor conditions. In particular, these guidelines address issues like fair prices, fair labor conditions, community development and environmental sustainability, etc. (Fairtrade International, 2014). Products adhering those guidelines are marked with the FT label.

FT products are often commodities such as coffee, chocolate, fruit juice, etc. (Fairtrade International, 2014). In particular, the coffee category has been heavily analyzed (Galarraga & Markandya, 2004; De Pelsmacker, Janssens, Sterckx, & Mielants, 2005; Arnot, Boxall, & Cash, 2006; Basu & Hicks, 2008; Trudel & Cotte, 2009; Cranfield, Henson, Northey, & Masakure, 2010; Tully & Winer, 2014) and primarily based on UK, US or Canadian data sets (Andorfer & Liebe, 2012, p. 427). These studies furnish empirical evidence that a FT label adds a utility surplus and therefore increases consumers' willingness-to-pay (WTP) for FT coffee.

Our research analyzes the German FT market for orange juice. This has several reasons: First, fairly traded fruit juice has not been a research focus so far (e. g., not a single study is cited in the meta-study of Tully & Winer, 2014). So our study contributes to the recent literature on FT. Second, orange juice is after apple juice the second most preferred fruit juice in Germany. While apples are grown in developed countries (e. g., Germany), where reasonable wages should be paid and therefore combinations with FT standards might be less important for consumers, oranges are primarily produced in emerging or poor countries (e. g., Brazil). Hence, the combination with a FT label should be effective for the orange juice category. Third, the German FT market is less examined so far but structurally similar to other FT markets in European countries and hence can serve as a blueprint.

Although the FT market in Germany is still a niche market, its potential should not be left unnoticed. Beside academic researchers in the field of consumer behavior, companies may also be interested in consumers' WTP for the FT label. The WTP can serve as a benchmark for consumers' value assigned to a fairly traded orange juice and may enter companies product portfolio planning (e.g., product differentiation, market segmentation, etc.). Within our study, we therefore focus on two research questions:

1. How high is the consumers' WTP for the FT attribute?
2. To what extend explain individual background variables differences in WTP across consumers?

To answer these questions, we conduct a discrete choice analysis (choice-based conjoint, see Rao, 2014 for details) and calculate respondents' WTP for the attribute "FT label". We estimate multinomial logit (MNL) and mixed logit (MXL) models. Furthermore, we account for observed consumer heterogeneity and determine the effect of demographic and psychographic characteristics on individuals' FT label WTP. In particular, we use the established consciousness for fair consumption (CFC) scale developed by Balderjahn, Peyer, & Paulssen (2013).

The next section describes the discrete choice model as well as the measurement model for the CFC construct. Section 3 contains the data description, the conjoint setup, and the empirical results of the study. Conclusions and advices for future research are given in section 4.

2 The Model

We determine respondents' WTP by an indirect-measurement approach using a discrete choice analysis (Rao, 2014). Several other techniques to estimate consumers' WTP exist which can be roughly divided into direct and indirect approaches. However, indirect approaches turned out to dominate direct approaches in several dimensions (e.g., flexibility and validity, see Breidert, Hahsler & Reutterer, 2006 for further details).
In a discrete choice analysis, it is assumed, that a choice alternative/product could be described by attributes (levels). A respondent evaluates several choice occasions with a fixed number of alternatives, respectively. In each choice occasion, it is assumed, that the

[1] Clausthal University of Technology, email: friederike.paetz@tu-clausthal.de
[2] Humboldt University Berlin, email: daniel.guhl@hu-berlin.de

respondent chooses that alternative, which maximizes his/her utility. We follow random utility theory and assume, that the utility of a respondent i for a certain alternative j in choice set t could be formulated as

$$u_{ijt} = x_{ijt} \cdot \beta_i + \alpha_i \cdot p_{ijt} + \varepsilon_{ijt}, \quad \varepsilon_{ijt} \sim EV(0,1), \tag{1}$$

where x_{ijt} is a dummy-coded vector of (non price-) attributes (here: brand, packaging and FT label), which describes the alternative, p_{ijt} is the price and β_i denotes the part-worth vector, while α_i is the (linear) price parameter of respondent i in case of heterogeneity. We expect α_i to be negative.

The first two additive terms in formula (1) constitutes the deterministic part of the utility. Since we consider the utility of a respondent as a random construct, we add an error term ε_{ijt} to the deterministic part. This error term captures all (random) effects, that are not included in the deterministic utility part, but also affect the utility of a respondent. E. g., A consumer intends to buy a certain product but is influenced by situation-specific factors at the point-of-sale (e. g., appearance of a friend, who persuades him/her to buy a different product) and buys something else. These factors are captured within the random part of the utility and cannot be estimated by the researcher. The specific error term distribution was chosen because in combination with the assumption that respondents are utility maximizers, the probability $pr_{it}(j^*)$ of choosing a specific alternative j^* given its deterministic utility can be written in closed-form. In particular, if ε_{ijt} is distributed type I extreme value, the classical MNL ("softmax-regression") model follows (Train, 2009):

$$p_{it}(j^*) = \frac{\exp\left(x_{ij^*t} \cdot \beta_i + \alpha_i \cdot p_{ij^*t}\right)}{\sum_{j=1}^{J} \exp\left(x_{ijt} \cdot \beta_i + \alpha_i \cdot p_{ijt}\right)}, \tag{2}$$

where J describes the number of alternatives in a choice set.

If the parameters are assumed to be the same for all respondents ($\beta_i = \beta$; $\alpha_i = \alpha$), the aggregated MNL model follows. If we take heterogeneity into account (MXL model), the individual parameters are specified as

$$[\beta_i,\ \alpha_i]' = \theta_i = \Gamma^T \cdot z_i + \nu_i, \quad \nu_i \sim \text{MVN}(0, \Omega). \tag{3}$$

The vector z_i includes an intercept and the individual-specific (background) variables (i. e., age, gender, as well as CFC) to capture observed heterogeneity. The matrix Γ contains the effects of those variables on respondents' preferences and unobserved heterogeneity is modeled by using a multivariate normal distribution with mean zero and a covariance matrix Ω (Allenby & Ginter, 1995).

Utility as a measure of preference over a set of alternatives is a latent construct and difficult to interpret and understand. To support decisions of (marketing) managers, it is common in quantitative marketing and economics to translate the rather abstract utility parameters into monetary units (i. e., WTP). WTP-values are readily interpretable and comparable across categories, studies, models, etc. To get the monetary equivalent of the utility for a certain product attribute (e. g. FT label vs. no FT label) we derive the marginal rate of substitution between the attribute and the price: $MRS = -(\partial u/\partial x)/(\partial u/\partial p)$. In case of a discrete attribute, $\partial u/\partial x$ simplifies to the utility difference with and without this attribute. Hence in our case the WTP of the FT label is (Tully & Winer, 2014):

$$\text{WTP}^{\text{FT}} = -\frac{\beta^{\text{FT}}}{\alpha}. \tag{4}$$

Please note, that if we take heterogeneity into account, WTP varies over respondents.

Based on the model assumptions it is straight forward to derive the likelihood function, which enables the estimation of the part-worth utility vectors. In case of absent unobserved heterogeneity, we maximize the log-likelihood function

$$LL(\beta, \alpha) = \sum_{i=1}^{I} \sum_{t=1}^{T} \sum_{j=1}^{J} \delta_{ijt} \cdot \ln\big(pr_{it}(j)\big). \tag{5}$$

The dummy-variable δ_{ijt} equals one if respondent i chooses alternative j in choice set t and zero otherwise.

In case of present unobserved heterogeneity, the log-likelihood function becomes

$$LL(\Gamma, \Omega) = \sum_{i=1}^{I} \ln\left(\int \prod_{t=1}^{T} \prod_{j=1}^{J} \big(p_{it}(j)\big)^{\delta_{ijt}} \phi(\nu) d\nu\right). \tag{6}$$

Because evaluating this multivariate integral is complex, we employ Monte Carlo simulation (Train, 2009). For each respondent we sample R draws of ν_i from the multivariate normal distribution ϕ and simulate the likelihood:

$$LL(\Gamma, \Omega) = \sum_{i=1}^{I} \ln\left(\frac{1}{R} \sum_{r=1}^{R} \prod_{t=1}^{T} \prod_{j=1}^{J} \big(p_{it}(j, \nu_{ir})\big)^{\delta_{ijt}}\right). \tag{7}$$

Respondents' CFC level is measured with an established 7-point rating scale comprising the following six items: (1) compliance with workers' rights, (2) freedom from forced labour, (3) abolition of illegal child labour, (4) non-discrimination in the workplace, (5) compliance with international statutory labour standards, and (6) fair wages for workers (see Balderjahn, Peyer and Paulssen, 2013, p. 548 for more details). CFC actually is a combination of a respondent's belief about the adherence to a labour standard and the personal importance attached to the adherence to this standard. The corresponding conceptual model is

$$\text{CFC}_i = \sum_{q=1}^{6} \text{Belief}_{iq} \times \text{Importance}_{iq}, \tag{8}$$

with i denoting (as before) the respondent and q the item. All six items are each combined with the phrases "I buy a product only if I believe that in its production ..." (belief component) and "How important is it for you personally that in companies ..." (importance component) (Balderjahn, Peyer & Paulssen, 2013, p. 548).

3 Empirical study

3.1 Data, conjoint-setup, and CFC construct

To answer our research questions, we conducted an empirical study and asked 138 respondents at two German universities to choose their preferred one-liter orange juice alternative in each of 16 choice sets, respectively (= 2208 choices). Each choice set contained a "no-choice" option and three orange juice alternatives. The alternatives were described by the attributes price (1.09 €/L, 1.39 €/L, 1.69 €/L, 1.99 €/L), brand (Albi, Granini, Hohes C, and Valensina, which are the top selling brands in Germany), type of packaging (PET (plastic), tetrapak (carton)), and the inclusion of a FT label (no/yes). For the price attribute we consider a linear specification and for the rest of the attributes we use dummy-coding (first attribute level = reference category). In addition, we estimated a parameter for the "no-choice"

item	believe × importance mean (sd)	factor loading ≥ 0.5	commu-nality ≥ 0.4	KMO index ≥ 0.5	item-to-total correlation ≥ 0.5	Cronbach's α (w/o item) ≥ 0.8
1	25.9 (13.0)	0.897	0.804	0.938	0.913	0.952
2	29.4 (13.5)	0.846	0.716	0.927	0.883	0.956
3	34.9 (14.2)	0.846	0.716	0.926	0.882	0.957
4	29.1 (13.3)	0.952	0.905	0.903	0.954	0.945
5	28.1 (14.0)	0.906	0.822	0.888	0.915	0.952
6	28.9 (14.2)	0.923	0.853	0.894	0.931	0.949
overall				0.912		0.960

Table 1. Results of an exploratory factor analysis of the CFC scale

utility. To incorporate observed heterogeneity, we used two dummy-coded variables, gender (female = 1) and age (young = 1, defined as age $\leq$ 25 years), and one numeric variable, CFC. All background variables were mean-centered (i. e., female = 0.493 and age = 0.610) to ease interpretation. Therefore, the estimated intercepts in Γ are the population means of the utility distributions.

The students (61.0 % 25 years or younger, 49.3 % female) participated voluntarily in our study. Most respondents were aware (aided awareness) of the brands Hohes C (91.3 %), followed by Granini (87.0 %), Valensina (59.4 %), and Albi (58.7 %). Almost all respondents were familiar with the FT label (93.5 %).

The first step in our analysis is estimating the CFC construct. Table 1 displays results of an exploratory factor analysis (EFA) for the CFC construct together with recommended threshold values indicating proper results. The sample-data correlation matrix is suited for FA, as the Kaiser-Meyer-Olkin (KMO) index reveals. The EFA results (i. e., high factor loadings, large average extracted variance (AEV)) support the unidimensionality assumption for the CFC-scale. Also, the scale performs well on internal consistency criteria (i. e., Cronbach's α, item-to-total correlation) attesting to its high reliability. In addition a confirmatory factor analysis (CFA) evaluates construct validity. This 6-item scale shows acceptable fit (standardized root mean square residual (SRMR) value $\leq$ 0.05, comparative fit index (CFI) $\geq$ 0.95). However, similar to Balderjahn, Peyer & Paulssen (2013, p. 549), the error terms of items 2 and 3 correlate rather strongly ($r(\epsilon_2, \epsilon_3) = 0.546$), so we follow their procedure and collapse both items into a single one by computing their average value before running the CFA. This improves model fit (SRMR = 0.023, CFI = 0.959), as well as the items' and scale's reliability, and the AEV increases to 0.84. In sum, our results are very similar to Balderjahn, Peyer & Paulssen (2013) and strongly support the CFC-scale's unidimensionality and reliability. These results nurture our belief that the scale is appropriate for measuring the CFC construct. Respondents' CFC-values were therefore estimated from a latent score regression based on the described CFA.

3.2 Discrete choice analysis

We estimated four different models to determine respondents' preferences for the considered attribute (levels) and in particular respondents' WTP for the FT label. Those models differ in their consideration of observed and unobserved heterogeneity. Table 2 informs about model characteristics and model fit statistics resulting from maximum (simulated) likelihood estimation with 2000 Halton draws (see Train, 2009).

MNL1 is the worst fitting model. Model MNL2 does account for observed heterogeneity and has an improved fit. Adding unobserved heterogeneity boosts model fit but also results in a lot more parameters. Model MXL2 outperforms all its nested versions. It is significantly better than MNL1, MNL2, and – most importantly – MXL1 ($LR = 78.803$, $p < 0.001$). The McFadden-R^2 is 0.486, also indicating a good model fit. Consequently, we will focus on model MXL2 for the rest of the paper.

Table 3 shows the average values of the population distribution of the utility parameters ($\bar{\gamma}$), the effects of the (mean-centered) background variables on the average utility parameters of the population sub-groups, and the square root of the diagonal elements in Ω as a measure of unobserved heterogeneity (ω). The average price parameter has an economically meaningful negative sign and a reasonable magnitude (i. e., elastic demand). The brands Granini, Hohes C, and Valensina possess a higher (mean) utility than the reference brand Albi. The carton packaging and the FT label increase utility, and hence preferences for products with these attributes are c.p. higher on average. Many background variables have significant effects, e. g., being older (2.555) or having a high level of CFC (1.045) decreases c. p. price sensitivity, while being younger (0.837) or having a high level of CFC (1.332) increases c.p. preference for FT products. Furthermore, a high level of CFC increases preference for the carton packaging. This may stem from a potential positive correlation between the CFC-construct and, e. g., "Consciousness for environmental friendly packaging". A high level of CFC also increases the utility of the "no-choice" option. Presumably these consumers have a higher likelihood to abstain from buying orange juice if no FT products are available. The gender variable does not strongly affect utility, and hence, we cannot find major gender effects. The ω-values reveal heterogeneous preferences, even after controlling for the background variables. E. g., the standard deviation (2.835) of the FT label's utility exceeds its mean value (2.418) indicating that, even though the mean is positive and reasonable in magnitude, the FT label does not have a positive utility for some people in the population ($\approx$ 19.5 %) (see Tully & Winer, 2014 for comparable results).

Figure 1 further illustrates the heterogeneity of preferences by depicting relative importances (Rao, 2014). The price attribute has the highest importance (46.7 %) on average (indicated by the black dot) followed by brand (24.0 %), FT label (15.9 %), and the packaging type (13.4 %). However, the distributions are wide implying unequal importances of attributes across respondents. On average the FT label is almost as important as the brand attribute. At the same time, for some people the FT label is not important at all. Again, this makes perfectly sense, because not all people are socially responsible (Tully & Winer, 2014). A good model (e. g., the MXL2 model) should be capable of differentiating between these (types of) consumers and in this way enable the company to calculate a realistic effect of a FT product introduction in order to exploit this heterogeneity.

	MNL1	MNL2	MXL1	MXL2
obs. het. (CFC, gender, age)	No	Yes	No	Yes
unobs. het. ($\Omega \neq 0$)	No	No	Yes	Yes
number of parameters	7	28	35	56
log-Likelihood	-2333.957	-2272.704	-1612.845	**-1573.443**
AIC	4681.915	4601.409	3295.689	**3258.886**
McFadden R^2	0.238	0.258	0.473	**0.486**

Table 2. Choice model comparison

parameter	$\bar{\gamma}$	γ_{CFC}	$\gamma_{\text{gender=female}}$	$\gamma_{\text{age}\leq 25}$	ω
price	-9.404**	1.045**	0.358	-2.555**	6.099**
Granini	0.605**	0.042	0.202	-0.010	3.166**
Hohes C	1.648**	-0.556**	0.505	0.788**	3.362**
Valensina	0.346*	0.068	0.857**	-1.049**	2.803**
carton	0.927**	0.587**	0.174	-0.049	3.712**
FT label	2.418**	1.332**	0.043	0.837**	2.835**
no-choice	-13.208**	2.966**	1.184*	-4.382**	10.843**

* indicates significance at $p < 0.10$; ** indicates significance at $p < 0.05$

Table 3. Selected parameter estimates of model MXL2

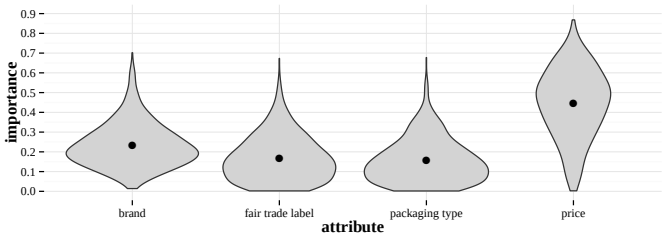


Figure 1. Densities of the relative importance of the four main attributes

In vein of our research questions and to simplify interpretation, we used the results of (the best performing) model MXL2 and calculated the WTP for each combination of background variable values based on parameters of the population distribution. We simulated draws from the estimated distributions (incl. estimation error) to integrate out the unobserved heterogeneity and to account for uncertainty in the estimates. The average WTP for the FT label is about 23 Eurocent, which implies a price premium of 17.4 % in the orange juice category. If only socially responsible persons are included in the WTP computation, the average WTP increases to 32 Eurocent (= 24.2 % price premium). These results are very close to those reported (e. g., 16.8 % in case all customers are included) in the meta-analysis of Tully & Winer (2014). Ignoring heterogeneity completely (MNL1) results in an underestimation of price sensitivity and we cannot differentiate anymore whether consumers are socially responsible, or not. In this case the WTP for the FT label is 32 Eurocent and hence upward biased by 39 %. So it is important to account for heterogeneity, not only because of model fit (see table 2) but also to get reasonable and unbiased estimates (Train, 2009).

As already mentioned in table 3, the WTP for the FT label differs because of the individual background variables. To analyze this in more detail, we also compute the WTP for the FT label for different sub-groups in the population (CFC categorized as low: CFC = -1, moderate: CFC = 0 (i. e., mean), and high: CFC = 1). While young women with a high level of CFC have the highest WTP for the FT label (41 Eurocent), older men with a low level of CFC are only willing to pay 5 Eurocent. Figure 2 contains further WTP results. The level of CFC appears to have the strongest WTP-effect. This has two reasons: A high level of CFC both increases preference for the FT label and decreases price sensitivity (see table 3). A younger age increases the FT label WTP also in general, and this effect is more pronounced for lower values of CFC. However, the effect of gender is small.

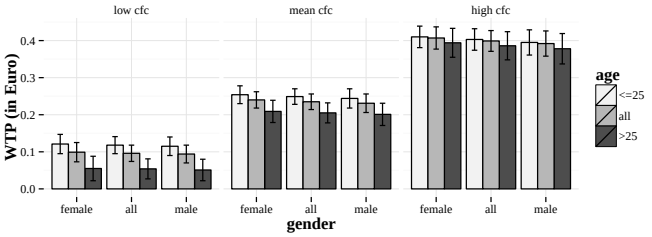


Figure 2. WTP for FT label in different population sub-groups

4 Conclusion

In order to answer our research questions regarding the consumers' WTP for the FT label and the individual background variables which affect the consumers' WTP for the FT label, we conducted an discrete choice experiment. We found evidence that the FT label increases the respondents' utilities and therefore their preference and WTP for an orange juice product. The average WTP for the FT label was about 23 Eurocent translating to a relative price premium of 17.4 % in the considered orange juice category. Furthermore, we observed considerable preference heterogeneity which translates into heterogeneity in individual WTP values. While young women who are highly conscious for fair consumption have the highest WTP, older men with a low level of CFC exhibit the smallest WTP for the

FT label. Hence, differences in the WTP for the FT label within the orange juice category can be explained by the consumers' individual background variables.

Our current study is limited in different ways: (1) We considered a student sample for convenience purposes (e.g., lower cost). Hence, it would be interesting to validate our results with a more general sample. (2) We assumed a normal distribution also for the price parameter, which allows for positive values even if the mean is negative. Hence, it might be worth considering using a (negative) log-normal distributed price parameter in the model (Daly, Hess, & Train, 2012) or estimate the model directly in WTP-space (Sonnier, Ainslie & Otter, 2007) and check whether the WTP results are robust.

Since this is one of the first studies which focuses on consumers' WTP for the FT label attribute in the orange juice category, there is a need for future research per se. We hope, that our results may serve as a basis for further analyses. E.g., in combination with FT label cost information and a certain competitive scenario, the application of our results within an equilibrium analysis may provide information for companies whether the adherence of FT standards could increase their profits.

REFERENCES

[1] Allenby, G. M. & Ginter, J. L. (1995). Using Extremes to Design Products and Segment Markets. *Journal of Marketing Research*, 32, 392-403.

[2] Andorfer, V. A. & Liebe, U. (2012). Research on Fair Trade Consumption – A Review. *Journal of Business Ethics*, 106, 415-435.

[3] Arnot, C., P.C. Boxall & Cash, S.B. (2006). Do ethical consumers care about price? A revealed preference analysis of Fair Trade Coffee purchases. *Canadian Journal of Agricultural Economics*, 54, 555-565.

[4] Basu, A. K. & Hicks, R. L. (2008). Label performance and the willingness to pay for Fair Trade coffee: A cross-national perspective. *International Journal of Consumer Studies*, 32, 470-478.

[5] Balderjahn, I., Peyer, M., & Paulssen, M. (2013). Consciousness for fair consumption: conceptualization, scale development and empirical validation. *International Journal of Consumer Studies*, 37, 546-555.

[6] Breidert, C., Hahsler, M., & Reutterer, T. 2006. A Review of Methods for Measuring Willingness-to-Pay. *Innovative Marketing*, 2, 8-32.

[7] Cranfield, J., Henson, S., Northey, J., & Masakure, O. (2010). An assessment of consumer preference for Fair Trade coffee in Toronto and Vancouver. *Agribusiness*, 26, 307-325.

[8] Daly, A., Hess, S., & Train, K. (2012). Assuring finite moments for willingness to pay in random coefficient models. *Transportation*, 39, 19-31.

[9] De Pelsmacker, P., Janssens, W., Sterckx, E., & Mielants, C. (2005). Consumer preference for the marketing of ethically labelled coffee. *International Marketing Review*, 22, 512-530.

[10] Fairtrade International (2014). Explanatory document for the Fairtrade standards for small producer organizations. Retrieved from: `http://www.fairtrade.net/fileadmin/user_upload/content/2009/standards/documents/2014-01-15_EN_SPO_Explan_Doc.pdf` (Last accessed: 20.08.2016).

[11] Galarraga, I. & Markandya, A. (2004). Economic techniques to estimate the demand for sustainable products: A case study for Fair Trade and organic coffee in the United Kingdom. *Economia Agraria y Recursos Naturales*, 4, 109-134.

[12] Howard, P. H. & Allen, P. (2008). Consumer willingness to pay for domestic fair trade: Evidence from the United States. *Renewable Agriculture and Food Systems*, 23, 242-253.

[13] Rao, V. R. (2014). *Applied Conjoint Analysis*. Springer.

[14] Sonnier, G., Ainslie, A., & Otter, T. (2007). Heterogeneity distributions of willingness-to-pay in choice models. *Quantitative Marketing and Economics*, 5, 313-331.

[15] Train, K. (2009). *Discrete choice methods with simulation*. Cambridge University Press.

[16] Trudel, R. & Cotte, J. (2009). Does it pay to be good? MIT Sloane Management Review, 50, 61-68.

[17] Tully, S. M. & Winer, R. S. (2014). The Role of the Beneficiary in Willingness to Pay for Socially Responsible Products: A Meta-analysis. *Journal of Retailing*, 90, 255-274.

ST-VIKOR: Accounting for Stochastic Data and Risk Attitudes in Multi-Criteria Decision Making

Madjid Tavana
Business Systems and Analytics Department. Distinguished Chair of Business Analytics
La Salle University, Philadelphia, PA 19141, USA.
E-mail: tavana@lasalle.edu; Web: http://tavana.us/
Business Information Systems Department. Faculty of Business Administration and Economics
University of Paderborn, D-33098 Paderborn, Germany

Debora Di Caprio
Department of Mathematics and Statistics. York University. Toronto, M3J 1P3, Canada
E-mail: dicaper@mathstat.yorku.ca
Polo Tecnologico IISS G. Galilei. Via Cadorna 14, 39100, Bolzano, Italy
E-mail: debora.dicaprio@istruzione.it

Francisco J. Santos-Arteaga
School of Economics and Management. Free University of Bolzano, Bolzano, Italy
E-mail: fsantosarteaga@unibz.it
Instituto Complutense de Estudios Internacionales. Universidad Complutense de Madrid, Spain
E-mail: fransant@ucm.es

Abstract

We define an extended version of the VIKOR multi-criteria decision making (MCDM) method that accounts for differences in the risk attitudes of the decision makers (DMs) when ranking stochastic alternatives. Our ST-VIKOR model is designed to solve MCDM problems characterized by stochastic data and DMs categorized by their risk averse or risk seeking behavior. These differences in risk attitudes determine the subjective beliefs of the DMs regarding the evaluation of each alternative per decision criterion and the resulting rankings. We present a case study in the banking industry to illustrate how differences in the risk attitudes of the DMs condition the rankings obtained. Moreover, we compare our results with those derived from a stochastic super-efficiency data envelopment analysis (DEA) model to demonstrate the efficacy of the method proposed. ST-VIKOR has many potential applications to diverse research areas ranging from economics to knowledge based and decision support systems.

Keywords: Multi-criteria decision making; VIKOR method; Stochastic data; Subjective judgments; Risk attitudes; Stochastic super-efficiency DEA.

1. The VIKOR method

The VIKOR method is a MCDM technique introduced by Opricovic (1998) that defines positive and negative ideal points and determines the relative distance of each alternative. After computing each relative distance, the method obtains a weighted compromise ranking determining the importance of each x_j alternative, with $j = 1,2,\dots,m$. The steps composing the compromise ranking algorithm are:

1. Define the rating functions f_{ij} that describe the value achieved by alternative x_j, $i = 1,2,\dots,n$, when considering the $i-th$ criterion. Compute the best, f_i^+, and the worst, f_i^-, values for all the rating functions. If the criterion considered is a positive one, then the corresponding extreme values are given by

$$f_i^+ = \max\left[(f_{ij}) \mid j = 1,2,\dots,m\right]; f_i^- = \min\left[(f_{ij}) \mid j = 1,2,\dots,m\right] \quad (1)$$

2. Calculate the values of S_j and R_j, $j = 1,2,\dots,m$, as follows

$$S_j = \sum_{i=1}^{n} w_i \frac{(f_i^+ - f_{ij})}{(f_i^+ - f_i^-)} \quad (2)$$

$$R_j = \max_i \left[w_i \frac{(f_i^+ - f_{ij})}{(f_i^+ - f_i^-)}\right] \quad (3)$$

where w_i are the weights reflecting the relative importance of the criteria. S_j represents the group utility measure and R_j the individual regret measure defined for each alternative x_j.

3. Compute the values of Q_j, $j = 1,2,\dots,m$, as follows

$$Q_j = v\left[\frac{(S_j - S^+)}{(S^- - S^+)}\right] + (1-v)\left[\frac{(R_j - R^+)}{(R^- - R^+)}\right] \quad (4)$$

where

$$S^+ = Min\left[\left(S_j\right) \mid j = 1,2,\dots,m\right]; S^- = Max\left[\left(S_j\right) \mid j = 1,2,\dots,m\right] \quad (5)$$

$$R^+ = Min\left[\left(R_j\right) \mid j = 1,2,\dots,m\right]; R^- = Max\left[\left(R_j\right) \mid j = 1,2,\dots,m\right] \quad (6)$$

and v is the subjectively defined weight supporting either maximum group utility $(v > 1/2)$, individual regret $(v < 1/2)$, or consensus between both strategies $(v = 1/2)$.

4. Rank the alternatives using the values of (S_j, R_j, Q_j), which give place to three different rankings that can be used to define and validate a compromise solution.

2. ST-VIKOR: decision making with stochastic data and different risk attitudes of DMs

We restate the model of Tavana et al. (2016) in order to provide some basic intuition and allow for direct comparisons based on the degree of risk aversion of the DM. These authors defined the following version of VIKOR so as to account for stochastic realizations of the different rating functions. Their model is based on the sample mean of the rating functions per criterion and alternative together with their associated coefficients of variation. Given a sample of l realizations, y_k, $k = 1,\dots,l$, the coefficient of variation (*cv*) is defined as $cv = \frac{\sigma}{\bar{y}}$, i.e. the ratio of the standard deviation, $\sigma = \sqrt{\frac{1}{l}\sum_{k=1}^{l}(y_k - \bar{y})^2}$, to the sample mean, $\bar{y} = \frac{\sum y_k}{l}$.

Consider the following decision matrix where the stochastic data are described in terms of rating means and coefficients of variation

	C_1	C_2	...	C_n
A_1	$[f_{11}, cv_{11}]$	$[f_{21}, cv_{21}]$	...	$[f_{n1}, cv_{n1}]$
A_2	$[f_{12}, cv_{12}]$	$[f_{22}, cv_{22}]$	...	$[f_{n2}, cv_{n2}]$
...	...	...	...	...
A_m	$[f_{1m}, cv_{1m}]$	$[f_{2m}, cv_{2m}]$	...	$[f_{nm}, cv_{nm}]$

$$W = [w_1, w_2, \dots, w_n]$$

$A_1, A_2, \dots, A_m$ represent the alternatives among which the DM has to choose, $C_1, C_2, \dots, C_n$ are the different criteria measuring the performance of the alternatives, and $w_i (i = 1,2,\dots,n)$ are the weights representing the relative importance assigned by the DM to each criterion. The f_{ij}

entries of the matrix correspond to the sample means of the rating functions for the $i-th$ criterion and alternative x_j, The coefficient of variation associated to the $i-th$ criterion and alternative x_j is denoted by cv_{ij}.

Tavana et al. (2016) determined the best f_i^+ and the worst f_i^- values for all the rating functions considering two different types of criteria, i.e. positive and negative. They used the maximum coefficient of variation obtained per criterion, i.e. $\max_j cv_{ij}$, in order to define the extremes of the rating functions as follows

$$f_i^+$$

Positive criterion $$max\ f_{ij}\ \times \left(1 + max\ cv_{ij}\right),\ \ \forall j\colon\ \ j = 1,2,\dots,m \tag{7}$$

Negative criterion $$min\ f_{ij}\ \times \left(1 - max\ cv_{ij}\right),\ \ \forall j\colon\ \ j = 1,2,\dots,m \tag{8}$$

$$f_i^-$$

Positive criterion $$min\ f_{ij}\ \times \left(1 - max\ cv_{ij}\right),\ \ \forall j\colon\ \ j = 1,2,\dots,m \tag{9}$$

Negative criterion $$max\ f_{ij}\ \times \left(1 + max\ cv_{ij}\right),\ \ \forall j\colon\ \ j = 1,2,\dots,m \tag{10}$$

and then computed the corresponding values of S_j and R_j, $j = 1,2,3,\dots,m$

$$S_j$$

Positive criterion $$\sum_{i=1}^{n} w_i \frac{max\ f_{ij}\ \times \left(1 + max\ cv_{ij}\right) - f_{ij}}{max\ f_{ij}\ \times (1 + max\ cv_i) - min\ f_{ij}\ \times (1 - max\ cv_i)} \tag{11}$$

Negative criterion $$\sum_{i=1}^{n} w_i \frac{minf_{ij}\ \times (1 - max\ cv_i) - f_{ij}}{minf_{ij}\ \times (1 - max\ cv_i) - max\ f_{ij}\ \times (1 + max\ cv_i)} \tag{12}$$

$$R_j$$

Positive criterion $$max_i \left\{ w_i \frac{max\ f_{ij}\ \times (1 + max\ cv_i) - f_{ij}}{max\ f_{ij}\ \times (1 + max\ cv_i) - min\ f_{ij}\ \times (1 - max\ cv_i)} \right\} \tag{13}$$

Negative criterion $$max_i \left\{ w_i \frac{minf_{ij}\ \times (1 - max\ cv_i) - f_{ij}}{minf_{ij}\ \times (1 - max\ cv_i) - max\ f_{ij}\ \times (1 + max\ cv_i)} \right\} \tag{14}$$

We should emphasize that one of the main differences between our model and that of Tavana et al. (2016) will be observed in the computation of the values of S_j, and R_j. In particular, note that the f_{ij} terms subtracted in the numerators of equations (11)-(14) do not depend on the coefficient of variation of the respective observations. Thus, the variability inherent to each alternative per criterion is not considered by the authors when computing the ranking positions. In the current paper, we account for this variability together with the effect that the risk attitude of the DMs has on the corresponding rankings.

2.1. The risk averse setting

The modifications implemented to the extended VIKOR model defined by Tavana et al. (2016) focus on the definitions of the best f_i^+ and the worst f_i^- values of the rating functions and the

resulting values of S_j and R_j. In all cases, we must differentiate the effects of positive and negative criteria in the respective definitions. Consider the limit values of the rating functions

$$f_i^+$$

Positive criterion $$max\ f_{ij}\ \times \left(1 - min_j\ cv_{ij}\right),\ \ \forall j\text{:}\quad j = 1,2,\dots,m \tag{15}$$

Negative criterion $$min\ f_{ij}\ \times \left(1 + min_j\ cv_{ij}\right),\ \ \forall j\text{:}\quad j = 1,2,\dots,m \tag{16}$$

$$f_i^-$$

Positive criterion $$min\ f_{ij}\ \times \left(1 - max_j\ cv_{ij}\right),\ \ \forall j\text{:}\quad j = 1,2,\dots,m \tag{17}$$

Negative criterion $$max\ f_{ij}\ \times \left(1 + max_j\ cv_{ij}\right),\ \ \forall j\text{:}\quad j = 1,2,\dots,m \tag{18}$$

As a measure of the spread exhibited by the realizations of a given variable, the coefficient of variation should condition the expected performance of a given alternative and, therefore, the rankings defined by the DMs. Thus, we introduce the coefficients of variation in the expectation terms defined by the DMs when evaluating the alternatives through the values of S_j and R_j for each $j = 1,2,\dots,m$. It should be emphasized that we will not require any additional information on the side of the DMs. The resulting rankings are determined by the evaluations received and their relative spreads.

We introduce expectations in the definitions of S_j and R_j using a set of Beta density functions computed by the DMs for different realizations of the rating functions and the coefficients of variation per alternative and criterion. The values of S_j, $j = 1,2,3,\dots,m$, for a negative criterion are given by

$$S_j$$

$$Beta\left(cv_{ij}\middle|\, max_j\ cv_{ij}\right)w_i$$

Negative criterion $$\sum_{i=1}^{n}\int_0^1 \left[\frac{min\ f_{ij}\ \times \left(1 + min_j\ cv_{ij}\right) - f_{ij} \times \left(1 + cv_{ij}\right)}{min\ f_{ij}\ \times \left(1 + min_j\ cv_{ij}\right) - max\ f_{ij}\ \times \left(1 + max_j\ cv_{ij}\right)}\right] x dx \tag{19}$$

$$Beta(x; cv_{ij}, \max_j cv_{ij}) = \frac{x^{cv_{ij}-1}(1-x)^{\max_j cv_{ij}-1}}{\int_0^1 u^{cv_{ij}-1}(1-u)^{\max_j cv_{ij}-1} du} \tag{20}$$

Note that we have included a new variable, x, in order to generate an expected value derived from the Beta density. This function has been designed to condition the expected value of S_j on the relative coefficients of variation of the corresponding ratings.

2.2.The risk seeker setting

In the risk averse case, the DMs preferred the alternatives exhibiting a lower coefficient of variation. In the current setting, the maximum coefficient of variation will be used to define the highest rating achievable by an alternative, while the minimum one determines the lowest rating. Thus, given a positive criterion, the alternative with the highest rating will only achieve the best f_i^+ value if it has also the highest coefficient of variation. The best and worst values of the rating function within the risk seeker setting are

$$f_i^+$$

Positive criterion $$max\ f_{ij}\ \times \left(1 + max_j\ cv_{ij}\right),\ \ \forall j\text{:}\quad j = 1,2,\dots,m \tag{21}$$

Negative criterion $$min\, f_{ij} \times \left(1 - max_j\, cv_{ij}\right),\ \forall j:\quad j = 1,2,\dots,m \qquad (22)$$

$$f_i^-$$

Positive criterion $$min\, f_{ij} \times \left(1 + min_j\, cv_{ij}\right),\ \forall j:\quad j = 1,2,\dots,m \qquad (23)$$

Negative criterion $$max\, f_{ij} \times \left(1 - min_j\, cv_{ij}\right),\ \forall j:\quad j = 1,2,\dots,m \qquad (24)$$

As in the risk averse case, when evaluating the different alternatives their volatility per decision criterion has to be accounted for together with the rating obtained. The corresponding values of S_j, $j = 1,2,3,\dots,m$, for a positive criterion are

$$S_j$$

Positive criterion $$\sum_{i=1}^{n} \int_0^1 \left[\frac{max\, f_{ij} \times \left(1 + max_j\, cv_{ij}\right) - f_{ij} \times \left(1 + cv_{ij}\right)}{max\, f_{ij} \times \left(1 + max_j\, cv_{ij}\right) - min\, f_{ij} \times \left(1 + min_j\, cv_{ij}\right)} \right] x dx \qquad (25)$$

$$Beta\left(min_j\, cv_{ij} \,|cv_{ij}\right) w_i$$

$$Beta(x; \min_j cv_{ij}, cv_{ij}) = \frac{x^{\min_j cv_{ij} - 1}(1-x)^{cv_{ij}-1}}{\int_0^1 u^{\min_j cv_{ij} - 1}(1-u)^{cv_{ij}-1} du} \qquad (26)$$

As with risk averters, a Beta density skewed towards its lower end is used to evaluate the more preferred alternatives. Equation (25) focuses on the inherent variability to define the corresponding Beta densities, with higher coefficients of variation constituting preferred choices.

2.3. Numerical Example

In this section, we analyze numerically the different rankings defined by the DMs depending on their attitudes towards risk. In order to do so, we generate ten random alternatives evaluated through five different criteria out of which the first three are positive and the last two negative. Each alternative is assigned ten random realizations per criterion, which are generated using a uniform distribution defined on the following domains: [0,5], [0,10], [0,15], [0,20] and [0,25]. Namely, the domain on which the realizations of the first criterion are defined is [0,5], that of the second criterion is [0,10], and so on.

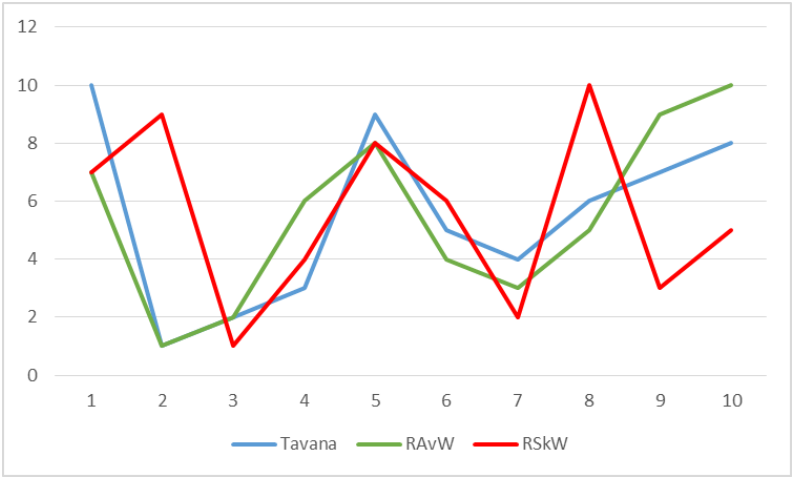


Figure 1. ST-VIKOR rankings determined by the risk attitudes of the DMs

The evaluations of the alternatives per VIKOR method assume a value of $v = 0.5$ when computing Q_j. Moreover, throughout the current numerical example we have implicitly assumed an identical weight for each criterion in order to focus on the effect that variability has on the rankings defined by the DMs. The following notation has been used to represent the different VIKOR models: Tavana refers to the model of Tavana et al. (2016), RAvW corresponds to the risk averse setting, while RSkW is the risk seeker case.

The resulting rankings are illustrated in Figure 1, where each alternative is represented on the horizontal axis and its corresponding ranking position on the vertical one. Table 1 reports the existing correlations among the different rankings. We observe how the risk seeker rankings differ substantially from the risk averse ones and the one of Tavana, which validates the differences observed in Figure 1. Thus, the risk attitude of the DMs has a direct effect on the resulting ranking of alternatives, which intensifies if the DMs condition their beliefs on the coefficients of variation when evaluating the corresponding alternatives.

Spearman's rho		Tavana	RAv	RSk
Tavana	Correlation Coefficient	1,000	,818**	,261
	Sig. (2-tailed)	.	,004	,467
	N	10	10	10
RAv	Correlation Coefficient	,818**	1,000	,030
	Sig. (2-tailed)	,004	.	,934
	N	10	10	10
RSk	Correlation Coefficient	,261	,030	1,000
	Sig. (2-tailed)	,467	,934	.
	N	10	10	10

Table 1. Ranking correlations among the different ST-VIKOR methods

3. Conclusion

ST-VIKOR allows the DMs to select the alternative that is more in accordance with their subjective preferences and risk attitudes while accounting for the uncertainty inherent to the real world when evaluating each alternative. Thus, if the DMs do not know the exact distribution from which the observations are drawn or are not neutral to the inherent risk, their resulting rankings will differ from the ones provided by more neutral models. Indeed, the proposed method is sufficiently flexible so as to account for both risky and uncertain environments and can be formally extended to other MCDM techniques such as PROMETHEE or TOPSIS.

Potential extensions of the current framework should consider the introduction of expected evaluations whose domain is determined by the entropy inherent to the data, strategic environments whose information structure is conditioned by the credibility of the reporters providing the data, and the emergence of interdependencies among the different decision criteria and their effect on the evaluations of the DMs.

References

Opricovic, S. (1998). Multicriteria Optimization of Civil Engineering Systems. Faculty of Civil Engineering, Belgrade.

Tavana, M., Kiani Mavi, R., Santos-Arteaga, F.J., & Rasti Doust, E. (2016) An Extended VIKOR Method Using Stochastic Data and Subjective Judgments. Computers and Industrial Engineering, Vol. 97, pp. 240–247.

Dominance based monte carlo algorithm for preference learning in the multi-criteria sorting problem: Theoretical properties

Tom Denat [1], Meltem Öztürk [2]

Abstract. In this article we study a new model-free Multi-Criteria Decision Aiding (MCDA) method for sorting problems where objects are assigned to predefined and ordered categories. The problem that we deal is the following: given a learning set of objects defined on multi-attributes and already assigned by the decision maker, how to find the assignments of the remaining objects. Being model-free, we do not assume that the decision maker's reasoning follows some well-known and explicitly described rules or logic system. We only assume that monotonicity should be respected as well as the learning set. The specificity of our approach is to be probabilistic. A Monte Carlo principle is used where the median operator aggregates the results of independent and randomized experiments. We proved that our final sortings respect the monotonicity and the learning set and the aggregation with the median operator converges almost surely.

1 Introduction

In this article we are studying a preference elicitation algorithm for the multi-criteria sorting problem, which consists in assigning objects to predefined and ordered categories. The classification of applications for bank credits into three classes -"acceptable", "to be discussed" and "rejected" - is a simple example of a sorting problem.

In the literature one can find different methods adapted to this problem, such as ELECTRE TRI ([Roy and Bouyssou, 1993]), rule based methods ([Greco et al., 2001]) or utility based methods ([Keeney and Raiffa, 1994]). For the decision maker, expressing her views on the parameters of these methods (such as weights, thresholds etc.) may sometime be a complex issue either because she does not understand completely the meaning of these parameters or because the necessary effort to understand the model is too important. For these reasons some authors developed some elicitation approaches where the decision maker provides only her expectations on the results such as "I think that the object a should be assigned to category 2, the object b should be assigned to category 1, etc." . Then these methods try to find the parameters that suit to this preference information (see for instance [Jacquet-Lagreze and Siskos, 1982], [Sobrie et al., 2013], [Greco et al., 2001]). The attractiveness of elicitation approaches comes from the fact that they only require an instinctive and subjective judgment that suits quite well to the decision makers expectations. Nevertheless there can be a very large number of parameters that suit to the expressed preferences or, at the opposite; it can be that no parameter would return these preferences. That could mean that the decision maker's preferences are not representable through the chosen model.

In this article we present a new sorting method based on an elicitation procedure with a learning set. Our method combines two properties that are not frequently met by the other methods for preference learning: a model free approach and a stochastic approach.

Being model-free means that we do not assume that the decision maker's reasoning follows some well known and explicitly described rules or logic system. The human judgment is probably too complex to be described by simple rules and may not be totally deterministic. In some contexts, in policy decision making for instance, such rules may be a good frame that helps the decision maker to build a coherent reasoning and some defendable argumentations about the accepted decisions. But in others it could be that none of the existing Multi Criteria Aggregation Procedure (MCAP) such as the ones cited on the previous paragraph can be rigorously defended. For instance when we classify instinctively companies into categories representing how globally responsible they are, it is possible that our intuitive reasoning is not based on an additive utility or a majority or logical rule but it is a mix of all of them with some additional noise. In our knowledge, there exist very few model-free methods for MCDA. ORCLASS ([Pinheiro et al., 2014]) is one of them. It is a model-free multi-criteria sorting method that includes an interaction procedure. This procedure is incremental and in each step, the aim is to find the best or the most appropriate object to add to the learning set in order to guarantee the monotonicity. Our method is inspired from ORCLASS but transforms it in a stochastic method where the interaction is limited to one step where the decision maker provides the full learning set.

Our method is based on a stochastic approach using a Monte Carlo procedure. It proposes a sorting that may be seen as a "center" of all the sortings that respect some expected properties, which are the monotonicity and the learning set. Many authors, mainly from psychology and behavior analysis ([Regenwetter et al., 2011], [Luce, 1995], [Carbone and Hey, 2000]), defend the idea that preferences are subject to randomness. There exist in the literature some MCDA methods using a probabilistic approach such as SMAA methods (Stochastic Multicriteria Acceptability Analysis, [Lahdelma et al., 1998]). However, SMAA methods are not model-free, they analysis the stability of the results of a given MCAP method (SMAA-TRI for ELECTRE TRI for instance, see [Tervonen et al., 2009]).

As we already mentioned, we want our sorting to satisfy the monotonicity and the learning set. Monotonicity is seen as a desirable property in MCDA and its violation is considered as pathological.

[1] Université Paris Dauphine, PSL Research University, CNRS, LAMSADE, 75016 Paris, France, denat@lamsade.dauphine.fr
[2] Université Paris Dauphine, PSL Research University, CNRS, LAMSADE, 75016 Paris, France, ozturk@lamsade.dauphine.fr

The respect of the learning set is also generally considered as a good property in MCDA (contrary to the machine learning). We say that a sorting respects the learning set if the resulting sorting respects the assignment given by the decision maker. Most of the elicitation methods for multi-criteria sorting respects the learning set as long as this learning set is compatible with their associated MCAP. Indeed, while in other multi-criteria methods these two concepts are only seen as good properties, in our algorithm, they are imposed and compose the only real frame of this method.

Our method, that we call Dominance Based Monte Carlo (DBMC) algorithm works as follows. The decision maker provides us a learning set (a subset of objects and their assignment), these objects are directly affected to their categories. Then, we choose randomly an object among the remaining ones and we assign it randomly to a category without violating monotonicity. This new assignment generates new constraints on the remaining assignments because of monotonicity. Then we choose another object randomly and assign it randomly to a category and so on and so forth until every object is fixed in a category. This random sorting will be considered as a trial. Obviously, randomness has an important impact in this sorting. In order to reduce this impact and to converge we make a large number of trials, probabilistically independents to each other. We collect the information about the consecutive results of the trials and we aggregate it in order to get a single sorting.

The article is organized as follows: Section 2 introduces notions and basic concepts, Section 3 presents the process of a trial, Section 4 shows how to aggregate the results of several trials, Section 5 is dedicated to several theoretical guarantees such as the convergence and the respect of monotonicity. Section 6 concludes the article with some perspectives.

2 Theoretical basis and notations

We define a sorting context as a $5-tuple\ S =< N, V, A, C, L >$:

- $N = \{1, ..., n\}$ is a finite set of criteria.
- Each criterion i is expressed on a discrete and finite scale v_i. V is the union of criterion scales.
- A is the set of objects to be sorted. Here it is considered that any combination of values on the criteria must be sorted i.e. $A = \prod_{i \in N} v_i$ and $|A| = m$.
- $C = \{1, 2, ..., r\} \subset \mathbb{N}$ is a set of r ordered categories in which the objects are to be sorted. We will thereafter assume that the higher a category is, the better it is.
- L represents the learning set: $L = < \Theta, f_l >$ where $\Theta \subseteq A$ is the subset of examples and $f_l : \Theta \to C$ is the assignment of these examples.
- The expected output of this problem is a sorting. A sorting is a function $f : A \to C$ that assign every possible object a to a category $f(a)$.

Monotonicity: Given two objects a, b, we say that a weakly dominates b , and denote by aDb, if a is at least as good as b on every criterion. We will thereafter simply use the term "dominates" to mention weak domination.

For any object $a \in A$ we call dominating cone of a, also denoted by $D^+(a) = \{a' \in A : a'Da\}$, the set of all objects that dominate a. For any object $a \in A$ we call dominated cone of a, also denoted by $D^-(a) = \{a' \in A : aDa'\}$, the set of all objects that are dominated by a. We say that a sorting f respects monotonicity if, for any $a, b \in A$ such as aDb, we have $f(a) \geq f(b)$.

The learning set: Given that we impose the respect of monotonicity and the learning set, the possible remaining assignments are then constraint to a smaller space. For instance, if $a'Da$ and aDa'' with a' already assigned to category 4 and a'' already assigned to category 2, then we can deduce that a can only be assigned to category 2, 3 or 4 (we can say that a can be assigned to the interval of categories $[2, 4]$). In order to introduce this notion of "interval assignment" not violating the monotonicity and respecting the learning set we define the notion of a "necessary interval sorting", also denoted by $\gamma : A \to \Delta$, (Δ being the set of category intervals). In other words this necessary assignment represents the fact that any object a will be assigned to a category at least as good as the best sorted object that is dominated by a and at most as good as the worst sorted object that weakly dominates a.

An object a is said fixed with an interval assignment γ if $\gamma_{min}(a) = \gamma_{max}(a)$.

Figure 1 shows the interval assignments that we may have in the beginning of DBMC algorithm. In this figure, we have two criteria and three categories. First criterion has 14 levels and the second one 9 levels, hence 126 objects must be assigned into three categories. As we see 10 objects belong to the learning sets (2 of them in category 1, 4 of them in category 2 and 4 of them in category 3). These ten assignments constrain the interval assignments of the remained objects : red ones in category 1, orange-colored ones in interval $[1, 2]$, yellow ones in category 2, light green ones in interval $[2, 3]$, ...

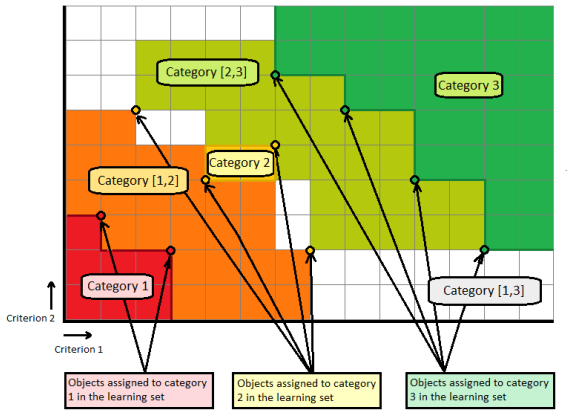


Figure 1. Illustration of the necessary interval sorting with two criteria, three categories and 10 objects in the learning set

The space of the possible sortings will obviously be empty if the learning set does not it self respect monotonicity. Therefore, from now on, we assume that the learning set respects monotonicity.

3 Process of a trial

In the DBMC algorithm each trial will be a random completion of the learning set that respects monotonicity. To do so, we iteratively choose uniformly a random object and assign it uniformly to a random category among the categories in which it could be sorted. Algorithm 1 presents this procedure.

We say that we add an information $< a, c >, a \in A, c \in C$ to the learning set when we impose the fact that the object a should be sorted in category c, i.e. $\Theta \leftarrow \Theta \cup \{a\}$ and $f_l(a) \leftarrow c$.

Algorithm 1: Random completion - trial

Data: Sorting context $S =< N, V, A, C, L >$
Result: Assignment $f_1 : A \to C$ monotonic and compatible with L

while $\exists a \in A$ *such that* $\gamma_{min}(a) \neq \gamma_{max}(a)$ **do**
  Choose randomly an object χ with a uniform distribution over A;
  Choose randomly a category Δ with a uniform discrete distribution between $\gamma_{min}(\chi)$ and $\gamma_{max}(\chi)$;
  Add the information $< \chi, \Delta >$ to the learning set;
  $\gamma_{max}(\chi) \leftarrow \Delta$;
  $\gamma_{min}(\chi) \leftarrow \Delta$;
  for $a^- \in D^-(\chi)$ **do**
    $\gamma_{max}(a^-) \leftarrow min\{\Delta, \gamma_{max}(a^-)\}$
  for $a^+ \in D^+(\chi)$ **do**
    $\gamma_{min}(a^+) \leftarrow max\{\Delta, \gamma_{min}(a^+)\}$

Properties of the random completion :
It seams quite obvious that the learning set is respected given that, by definition, these assignments are imposed from the beginning. We show in the following that the obtained sorting respects monotonicity condition.

Proposition 1. *If we assume that the learning set respects the monotonicity, then the sorting f_1 obtained by Algorithm 1 respects also the monotonicity.*

Proof : We will show that at each step, the interval assignment γ verifies $\forall a, b \in A$ if aDb then $\gamma_{max}(a) \geq \gamma_{max}(b)$, $\gamma_{min}(a) \geq \gamma_{min}(b)$.

Let us assume that at a given step St the interval assignment respects these properties. Let us now assume that we add the object $a \in A$ in category c such that $\gamma_{min}(a) \leq c \leq \gamma_{max}(a)$ (bounds of the object a before it was added to the learning set).
Now let a^+ and a^- be two objects of A such as $a^+ D a^-$, we have different possibilities:

i. $a^+, a^- \in A \backslash (D^-(a) \cup D^+(a))$: nothing changed in γ for a^+ and a^- and there couldn't be new violation of monotonicity about these objects nor could $\gamma_{min}(a^+)$ become higher than $\gamma_{max}(a^+)$.
ii. $a^+ \in A \backslash (D^-(a) \cup D^+(a))$ and $a^- \in D^+(a)$: not possible given that $a^+ D a^-$.
iii. $a^- \in A \backslash (D^-(a) \cup D^+(a))$ and $a^+ \in D^-(a)$: not possible given that $a^+ D a^-$.
iv. $a^- \in A \backslash (D^-(a) \cup D^+(a))$ and $a^+ \in D^+(a)$: nothing changes for $\gamma_{max}(a^+)$, $\gamma_{min}(a^-)$ and $\gamma_{max}(a^-)$. $\gamma_{min}(a^+)$ might increase so no new violation of the monotonicity with respect to $\gamma_{min}(a^-)$. If $\gamma_{min}(a^+)$ increases then, $\gamma_{min}(a^+) = c \leq \gamma_{max}(a) \leq \gamma_{max}(a^+)$ then we still have $\gamma_{max}(a^+) \geq \gamma_{min}(a^+)$.
v. $a^+ \in A \backslash (D^-(a) \cup D^+(a))$ and $a^- \in D^-(a)$: similar to the item iv.
vi. $a^+, a^- \in D^+(a)$ then $\gamma_{min}(a^+) \leftarrow max\{\gamma_{min}(a^+), c\}$ while $\gamma_{min}(a^-) \leftarrow max\{\gamma_{min}(a^-), c\}$. Initially $\gamma_{min}(a^+) \geq \gamma_{min}(a^-)$ then $max\{\gamma_{min}(a^+), c\} \geq max\{\gamma_{min}(a^-), c\}$. Moreover $c \leq \gamma_{max}(a) \leq \gamma_{max}(a^+)$ then $\gamma_{max}(a^+) \geq \gamma_{min}(a^+)$ (same with a^-).
vii. $a^+, a^- \in D^-(a)$: similar to the item vi.

Thus these two properties are still respected at the step $St + 1$. If the learning set is monotonic then the interval assignment at the beginning of the process is the same as if all the elements of the learning set were added successively to an empty learning set. The initial interval assignment as well as the following ones and thus the output sorting of the random completion will then by recurrence also respect these two properties. ■

Distribution of a trial :
As we just saw, the result of a trial is subject to randomness. Thus, the question arose to know whether or not we can find its distribution. It appeared to us that the distribution of a trial could theoretically be found modelling this problem with the use of a markov chain where the state space would be all the possible learning sets that are compatible with the original learning set and that are monotonic. Indeed, at each step of the algorithm, the probability distribution of the next step only depends on the current step. This markov chain is not positive recurrent given that the learning set increases at each step and thus there cannot be no stationary probability distribution on it. Moreover, the transition matrix seems very complex to model given the number of possible learning sets. A fortiori so does the calculation of the limit of this matrix power T when T tends to infinity. Nevertheless, even not knowing exactly this distribution, some problems can be solved. We wanted to know if every sorting that respects monotonicity and the learning set could be returned with a probability higher than 0. We are about to prove that the answer to this question is "yes".

Proposition 2. *While doing a random completion, any sorting that is compatible with the current necessary interval sorting and respects monotonicity can be chosen with a probability higher than $\frac{1}{r^m}$ (r being the number of categories and m being the number of objects).*

Proof : Let us denote by $\{a_1, a_2, ..., a_m\}$ the permutation of objects of A representing the order in which these objects are selected by Algorithm 1 and let us consider a sorting f_0. Then, given that during a random completion, at each one of the m steps, an object a is assigned randomly to a category with a uniform distribution among $C' \subseteq C$ and $|C| = r$, the probability that the sorting f obtained by the random completion is equal to f_0 must be higher than $\frac{1}{r^m}$. Formally $\mathbb{P}(f(a_1) = f_0(a_1) \cap f(a_2) = f_0(a_2) \cap ... \cap f(a_m) = f_0(a_m))$ $= \mathbb{P}(f(a_1) = f_0(a_1)) \times \mathbb{P}(f(a_2) = f_0(a_2) | f(a_1) = f_0(a_1)) \times ... \times \mathbb{P}(f(a_m) = f_0(a_m) | f(a_1) = f_0(a_1) \cap ... \cap f(a_{m-1}) = f_0(a_{m-1})) \geq \frac{1}{r^m}$. ■

We also wondered if every sorting that respects monotonicity and the learning set would have the same probability to be returned. For this, we analysed first of all a very special case which has an empty learning set and where the first chosen object of Algorithm 1 is an object having the maximum value in all the criteria (dominating object).

Lemma 1. *If the learning set is empty then, the probability for all the objects to be sorted in category 1 is strictly higher than $(m \times r)^{-1}$*

Proof : If at the first step the dominating object is chosen and the category 1 is attributed then all the other objects will be sorted in category 1. ■

This lemma shows us that all the assignments do not have the same probability to be found with Algorithm 1.

Proposition 3. *While doing a trial in DBMC from an empty learning, all the still possible sortings do not have the same probability to be obtained at the end of the trial.*

Proof : The number of possible assignments with no training set is strictly higher than $(m \times r)$, thus all the possible assignments cannot have a probability to be returned equal or higher than $(m \times r)^{-1}$. ■

It might have been preferable to use a uniform distribution on the possible sortings for the trials. However, modelling a uniform distribution over the possible sortings would probably require to list all of them first which given their number would make it not applicable from either time or space computational point of view.

Computational complexity of a trial :

The number of steps in a trial can not be higher than m. Furthermore, at each step we modify the necessary interval sorting γ (γ_{min} and γ_{max}) at most $|D^+(a) \cup D^-(a)| \leq m$ times. Thus, the computational complexity of a trial is in $O(m^2)$.

4 Collecting and aggregating the trial's information

To apply the DBMC algorithm, we first complete randomly T times the original sorting context. Then, for every object, we note in a DBMC statistic vector in what category it has been assigned at every trial. Finally, we aggregate these vectors to assign each object to the category in which it has "globally" been assigned during the T trials.

Definition 1 (DBMC Statistic)**.** *We call a DBMC Statistic of S, $\varphi : A \times \mathbb{N} \rightarrow C^T$ the vector in which we store the results of T random completions of S such as $\forall T \in \mathbb{N}, a \in A, k \in [1;T]$, $\varphi_k(a,T)$ represents the category in which a has been assigned at the k^{th} trial among T trials.*

Formally we get a DBMC Statistic of S as it is explained in Algorithm 2.

Algorithm 2: Building a DBMC Statistic

Data: Sorting context $S =< N, V, A, C, P >$
Result: DBMC Statistic, $\varphi : A \times \mathbb{N} \rightarrow \mathbb{N}^T$
for j *from* 1 *to* T **do**
  $S' \leftarrow S$;
  Complete randomly S' using Algorithm 1;
  for $a \in A$ **do**
    $\varphi_j(a,T) \leftarrow \gamma^{S'}_{min}(a)$ or $\gamma^{S'}_{max}(a)$;

The DBMC Statistic contains all the information about the results of T trials. In order to have a single result about the assignment of objects of A, we need to aggregate this information knowing that different aggregation measures can be used. In statistic such aggregations are generally related to the measures of central tendency. A measure of central tendency ([Weisberg, 1992] [McCluskey and Lalkhen, 2007]) is an operator whose function is to synthesize the information of a vector (z) identifying "the central position" within that set of information. These measures ($\Gamma : \mathbb{N}^T \rightarrow \mathbb{N}$) respect some conditions:

- *boundary condition* : $min(z) \leq \Gamma(z) \leq max(z)$
- *symmetry condition*: $\Gamma(z_1, z_2, \ldots, z_T) = \Gamma(z_{\pi(1)}, z_{\pi(2)} \ldots, z_{\pi(T)})$ where π is a permutation map.

We define a DBMC Sorting the results obtained on the DBMC statistic by the use of an aggregation operator.

Definition 2 (DBMC Sorting)**.** *Let us call a DBMC Sorting DCl : $A \times \mathbb{N} \rightarrow C$ such as $DCl(a,T) = \Gamma(\varphi(a,T))$, where φ is a DBMC statistic, Γ an aggregation operator and T number of trials.*

In the literature, probably the three mostly used central tendency measures are the mode, the arithmetic mean and the median.

- The mode, here denoted by $Mode(z)$ returns the number that was found the more often in the vector z.

We can observe that both the median and the arithmetic mean are particular cases of a more general measure called OWA (ordered weighted averaging) operators introduced by Yager ([Yager, 1988]). An OWA operator of dimension T is a mapping $OWA : \mathbb{R}^T \rightarrow \mathbb{R}$ such that $OWA(z_1, \ldots, z_T) = \sum_{j=1}^{T} w_j b_j$ where b_j is the j^{th} largest of the z_i. $W = [w_1, \ldots, w_T]$ is an associated collection of weights lying in the unit interval and summing to one. OWA operators are known to be monotone, i.e. if $z'_i > z_i$ then $OWA(z_1, ..., z'_i, ..., z_T) \geq OWA(z_1, ..., z_i, ..., z_T)$. As mentioned earlier, the arithmetic mean and the median are two specific examples of OWA operators.

- The arithmetic mean ($Average(z)$) is an OWA with $W = [\frac{1}{T}, \frac{1}{T}, ..., \frac{1}{T}]$.
- The median ($Median(z)$) is an OWA with $W_{\frac{T}{2}} = 1$ and $W_j = 0, \forall j \neq \frac{T}{2}$.

We present in the following some notations about the use of such measures in DBMC algorithm.

- Let us denote by $\Lambda(a,T)$ the random variable that represents the sorting of the object a by the ModeDBMC algorithm with T trials. Formally $\Lambda(a,T) = Mode(\varphi(a,T))$
- Let us denote by $\eta(a,T)$ the random variable that represents the sorting of the object a by the MedianDBMC algorithm with T trials. Formally $\eta(a,T) = Median(\varphi(a,T))$
- Let us denote by $\delta(a,T)$ the random variable that represents the sorting of the object a by the AverageDBMC algorithm with T trials. Formally $\delta(a,T) = Average(\varphi(a,T))$

For obvious reasons these operators both verify the boundary and the symmetry conditions. The mean, median and mode are all valid measures of central tendency, but under different conditions, some measures of central tendency become more appropriate to use than others. For instance the AverageDBMC algorithm induces a sum between the values of the categories in C. If the method is used in a context in which the categories have a cardinal meaning it could eventually make sense. However, in sorting problems the categories generally have an ordinal meaning and then adding the values representing the categories (that could even be defined without the use of numerical values such as "acceptable", "to be discussed" and "refused") makes no sense. By the way this remark could be done about any OWA operator that would involve a vector W with more than one strictly positive value, given that it would induce a sort of weighted sum. Thus, the only OWA operators that may be accepted in a sorting problem with ordinal categories is an OWA operator with only a 1 and $T-1$ 0's which is equivalent to different percentiles. Given that we do not want to be particularly optimistic nor pessimistic, it seems to us more legitimate to choose the median value. Let us remark that in a two categories sorting context these three variants would be equivalent.

5 Expected properties

There are some good properties that we would like our method to fulfill. We want the result of our DBMC respects the learning set and monotonicity. Also, we would like to know that, when the number of trials grows, the result of the method becomes less aleatory.

Respect of the learning set:

Proposition 4. *Any DBMC Sorting using a central tendency measure respects the learning set.*

Proof : Let $L = < \Theta, f_l >$ be the learning set with $a \in \Theta$ and $f_l(a) = c$. Since at each trial the learning set is respected, we have $\varphi(a, T) = (c, \ldots, c)$. All central tendency operators Γ satisfying the boundary condition ($min(z) \leq \Gamma(z) \leq max(z)$) the result of DBMC Sorting will be $DCl(a, T) = \Gamma(\varphi(a, T)) = c$. ■

Monotonicity:

Proposition 5. *Any DBMC Sorting using an OWA operator respects monotonicity.*

Proof : Let us consider $a, b \in A$ such as aDb. Since every random completion is monotonic, we have $\varphi(a, T) \geq \varphi(b, T)$. By the monotonicty of OWA operators ([Yager, 1988]), we have $\Gamma(\varphi(a, T)) \geq \Gamma(\varphi(b, T))$, hence we have $DCl(a, T) \geq DCl(b, T)$. ■

We can then conclude that MedianDBMC algorithm and AverageDBMC algorithm respect the monotonicity. However, we will show in the following that it is not the case with the ModeDBMC algorithm where the monotonicity is broadly violated when the learning set is small.

Proposition 6. *ModeDBMC algorithm does not guarantee the respect of monotonicity.*

Proof : Let us give a theoretical example where monotonicity is not respected with ModeDBMC algorithm.
Let us assume that we have a sorting problem with four categories (c_1 being the worst one) and that we apply our Algorithm 1 eleven times. Let a_0, a_1 be two objects that are not included in the learning set, such that a_1 dominates a_0. Let us assume that after running the algorithm with 11 trials , a_0 and a_1 are assigned as illustrated in Table 1 and Table 2. This feature is possible, for instance if at each trial a_0 and a_1 are the two objects that are chosen first. As we can see at the second and at the fifth trials the object a_1 is one category higher than a_0 due to the fact that it dominates it. However the object a_0 was sorted with the highest frequency in category 3 (5 times) while the object a_1 was assigned with the highest frequency in category 1 (4 times). Thus, a_0 would be sorted better than the a_1 with the ModeDBMC procedure.

Trial	1	2	3	4	5	6	7	8	9	10	11	Mode
a_0	c_3	c_3	c_2	c_1	c_3	c_1	c_1	c_3	c_1	c_3	c_2	c_3
a_1	c_3	c_4	c_2	c_1	c_4	c_1	c_1	c_3	c_1	c_3	c_2	c_1

Table 1. Results of the 11 trials for a_0 and a_1. The square (a_i, j) represents the category in which a_i is affected in the jth trial.

	c_1	c_2	c_3	c_4
a_0	4/11	2/11	5/11	0/11
a_1	4/11	2/11	3/11	2/11

Table 2. Frequency of sorting for each object in each category

■

Given that AverageDBMC has no meaning in an ordinal context and that ModeDBMC does not guarantee the monotonicity, we will only consider MedianDBMC in the rest of the article.

Convergence :

We would like to know that, when the number of trials grows, the result of DBMC Sorting becomes less aleatory. Thus we will later prove that the result of the MedianDBMC converges almost surely when T tends to infinity.

We say that the sequence X_T converges almost surely towards X when $T \to \infty$ if $\mathbb{P}(\lim_{T\to\infty} X_T = X) = 1$. Almost sure convergence implies convergence in probability (by Fatou's lemma) and thus is considered as a strong convergence (although it does not necessarily implies convergence in mean). Let us remark that by definition of Algorithm 2, it is considered that $\varphi_k(a, T)$ (category in which the object a is sorted at the k^{th} trial) are i.i.d.

In order to demonstrate the convergence, we need to introduce some notations.

Let $\rho_{a,c} = \mathbb{P}(\varphi_k(a, T) = c)$ be the probability that the object a is sorted in category c.

Let $Nb_{a,c,T}$ be the number of time that a was sorted in category c among T trials. In fact, $Nb_{a,c,T}$ stores the information of DBMC statistic into a smaller vector. Let us remark that while computing our algorithm for DBMC Sorting we preferred to use $Nb_{a,c,T}$ instead of DBMC statistics since $Nb_{a,c,T}$ requires less memory space and facilitates the application of the algorithm since the order of the assessments in $\varphi(a, T)$ is not important for MedianDBMC.

We thereafter demonstrate that MedianDBMC ($\eta(a, T)$) almost surely converges to the lowest category $c \in C$ such as $\mathbb{P}(\varphi_k(a, T) \leq c) \geq \frac{1}{2}$ as $T \to \infty$.

Proposition 7. *Let us call ϵ_a the lowest category such as $\mathbb{P}(\varphi_k(a, T) \leq \epsilon_a) \geq \frac{1}{2}$. $\eta_{a,T}$ converges to ϵ_a a.s when $T \to \infty$.*

Proof : By definition $\eta(a, T)$ is equal to the lowest category $c' \in C$ such as $\sum_{k=1}^{T} 1_{\{\varphi_k(a,T) \leq c'\}} \geq \frac{1}{2} \times T$. Thus, for any $c \in C$, we have:

$$\mathbb{P}(\eta(a, T) > c) = \mathbb{P}(\sum_{k=1}^{T} 1_{\{\varphi_k(a,T) \leq c\}} < \frac{1}{2} \times T)$$

Then, $\sum_{k=1}^{T} 1_{\{\varphi_k(a,T) \leq c\}} \sim \beta(\alpha_c, T)$ with $\alpha_c = \mathbb{P}(\varphi_k(a, T) \leq c)$ and β being the standard notation for the binomial distribution with parameters α_c, T.

We now use de Moivre-Laplace theorem, Φ being the standard notation for the cumulative distribution function of the standard normal distribution :

$\mathbb{P}(\sum_{k=1}^{T} 1_{\{\varphi_k(a,T) \leq c\}} < \frac{1}{2} \times T) \to \Phi(\frac{\frac{1}{2} \times T - T \times \alpha_c}{\sqrt{T \times \alpha_c \times (1-\alpha_c)}})$
since
$\Phi(\frac{\frac{1}{2} \times T - T \times \alpha_c}{\sqrt{T \times \alpha_c \times (1-\alpha_c)}}) = \Phi(\frac{\sqrt{T} \times (\frac{1}{2} - \alpha_c)}{\sqrt{\alpha_c \times (1-\alpha_c)}})$

we can conclude that
$\mathbb{P}(\sum_{k=1}^{T} 1_{\{\varphi_k(a,T)\leq c\}} < \frac{1}{2}T) \to \Phi(\frac{\sqrt{T}\times(\frac{1}{2}-\alpha_c)}{\sqrt{\alpha_c\times(1-\alpha_c)}})$ when $T \to \infty$
Thus $\frac{1}{2} - \alpha_c > 0 \Rightarrow \eta(a,T) > c \quad a.s$ and $\frac{1}{2} - \alpha_c < 0 \Rightarrow \eta(a,T) \leq c \quad a.s$
$\Rightarrow \eta(a,T) > \epsilon_a - 1$ and $\eta(a,T) \leq \epsilon_a \; a.s$
$\Rightarrow \eta(a,T) = \epsilon_a$. ■

6 Conclusion

We presented a preference elicitation algorithm for sorting based on a Monte Carlo principle and we proved that it meets some nice properties: the respect of monotonicity and the learning set and the convergence of the median operator. Being model-free, the DBMC algorithm can be seen by some decision makers as a black box. Surely, this method may be not adapted for some contexts in which some argumentations are needed such as policy decision making. However, this point may also have some advantages. The absence of a multicriteria aggregation procedure permeates dealing with any learning set as long as the monotonicity is respected (for instance veto phenomenon or some special types of dependence between criteria can be handled in our approach) and which may be a good point in complex and/or intuitive contexts where the human reasoning is not known and difficult to access. In order to better understand the contexts adaptable to our algorithm we are preparing some experiments. We will do tests using k-fold validation with real data (coming from UCI repository) in order to compare the results of DBMC with several MCAP such as MR-SORT (elicitation for simplified ELECTRE TRI), UTADIS (elicitation for additive utility model), DRSA (elicitation for rule based models).

Some remarks about when to use our algorithm: Due to its functioning, this algorithm requires as an input the use of discrete finite scales on the criteria. In many decision problems some criteria may be expressed on continuous scales such as a number of square meters, or a price. A possibility may be to discretize the continuous values. But this discretization might create a loss of information and could be subject to controversy if not operated carefully. Furthermore, the computational complexity of this algorithm being proportional to the square of the number of the objects, if the number of criteria or the number of classes in scales is too high this complexity might make the algorithm not runnable (which should be taken into account if some continuous scales are to be discretized). DBMC seems particularly adapted to multi-criteria sorting problems with criteria expressed on discrete scales with a limited number of objects. As an example we ran our algorithm with 100.000 combinations of criteria and 100 shots within approximately 2 hours and a half. Hence, from this dimensional point of view it may be suitable to a relatively large panel of multi-criteria sorting problems such as for example problems with 7 criteria expressed on scales of 5 categories. DBMC requires the learning set to respect the monotonicity. In practice, while interviewing decision makers, violations of monotonicity are very frequent. In such cases these violations must be treated before the beginning of the procedure. Different solutions can be proposed: deleting some objects from the learning set or making the affectation of some objects more flexible using intervals of categories. For the first solution, we have to decide which objects must be deleted. The most radical conclusion will be to delete all the objects causing a violation of monotonicity. Intermediate solution can be the cancellation of the minimum numbers of objects from the learning set in order to satisfy the monotonicity. This problem is one of the future works of our study. Concerning the second solution, the idea is simple. When the assignments of a couple a and b violates the monotonicity (aDb, a affected in C_i and b in C_j with $C_i < C_j$), we can consider that a and b are affected to the smallest interval of categories containing their initial categories (a and b affected in the interval $[C_i, C_j]$).

As a conclusion, the processing of the violation of monotonicity and the preparation of experiments with real data sets are the perspectives of our study.

Acknowledgement The authors would like to thank the anonymous reviewers for their helpful and constructive comments that contributed to improving the final version of the paper.

REFERENCES

[Carbone and Hey, 2000] Carbone, E. and Hey, J. D. (2000). Which error story is best? *Journal of Risk and Uncertainty*, 20(2):161–176.

[Greco et al., 2001] Greco, S., Matarazzo, B., and Slowinski, R. (2001). Rough sets theory for multicriteria decision analysis. *European Journal of Operational Research*, 129(1):1–47.

[Jacquet-Lagreze and Siskos, 1982] Jacquet-Lagreze, E. and Siskos, J. (1982). Assessing a set of additive utility functions for multicriteria decision-making, the {UTA} method. *European Journal of Operational Research*, 10(2):151 – 164.

[Keeney and Raiffa, 1994] Keeney, R. and Raiffa, H. (1994). Decisions with multiple objectives preferences and value tradeoffs, cambridge university press, cambridge, new york, 1993, 569 pages, isbn 0-521-44185-4 (hardback), 0-521-43883-7 (paperback). *Behavioral Science*, 39(2):169–170.

[Lahdelma et al., 1998] Lahdelma, R., Hokkanen, J., and Salminen, P. (1998). SMAA - stochastic multiobjective acceptability analysis. *European Journal of Operational Research*, 106(1):137–143.

[Luce, 1995] Luce, R. D. (1995). Four tensions concerning mathematical modeling in psychology. *Annual Review of Psychology*, 46(1):1–27.

[McCluskey and Lalkhen, 2007] McCluskey, A. and Lalkhen, A. G. (2007). Statistics ii: Central tendency and spread of data. *Continuing Education in Anaesthesia, Critical Care & Pain*, 7(4):127–130.

[Pinheiro et al., 2014] Pinheiro, P. R., Machado, T. C. S., and Tamanini (2014). Handing the classication methodology orclass by tool orclassweb. *Intelligent Computing Theory*, pages 61–72.

[Regenwetter et al., 2011] Regenwetter, M., Dana, J., and Davis-Stober, C. P. (2011). Transitivity of preferences. *Psychological Review*, 118(1):42.

[Roy and Bouyssou, 1993] Roy, B. and Bouyssou, D. (1993). *Aide Multicritère à la Décision : Méthodes et Cas*. Economica, Paris.

[Sobrie et al., 2013] Sobrie, O., Mousseau, V., and Pirlot, M. (2013). Learning a majority rule model from large sets of assignment examples. In *Algorithmic Decision Theory*, pages 336–350. Springer.

[Tervonen et al., 2009] Tervonen, T., Figueira, J. R., Lahdelma, R., Dias, J. A., and Salminen, P. (2009). A stochastic method for robustness analysis in sorting problems. *European Journal of Operational Research*, 192(1):236–242.

[Weisberg, 1992] Weisberg, H. (1992). Central tendency and variability sage university paper series on quantitative application in social sciences, series no. 07–038. a. virding, ed.

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

Parallel personalized pairwise learning to rank

Murat Yağcı[1] and **Tevfik Aytekin**[2] and **Hürol Türen**[3] and **Fikret Gürgen**[4]

Abstract. Methods based on online matrix factorization are commonly used for personalized learning to rank (LtR). These methods often use stochastic gradient descent (SGD) for optimization. SGD has a sequential nature which can be problematic for large relevance feedback datasets due to late convergence or when multiple passes are required over the dataset. In this paper, we investigate a shared memory lock-free parallel SGD scheme to deal with these problems in a *personalized pairwise LtR* setting. We show that such a parallel online algorithm is directly applicable to pairwise LtR as in pointwise LtR. We also show that a few modifications to the parallel online algorithm help solve several problems arising from stream processing of implicit user feedback. Experiments with the two proposed algorithms show remarkable results with respect to their comparative ranking ability and speedup patterns.

1 Introduction

Personalized LtR [7, 10, 16] refers to machine learning methods which build contextual ranking models, where the user is part of the context. Methods based on online matrix factorization are commonly used for personalized LtR, where typically a low-rank factor model is learned to predict unknown preferences of a user using known preferences in the system. Personalized LtR can be classified as pointwise [8, 6], pairwise [15, 14], and listwise [17] LtR. Basically, assuming that the user is the only context, these methods work on $(user, item)$, $(user, item_1, item_2)$, and $\left(user, item_1, \ldots, item_{|L|}\right)$ tuples respectively, where L is a list of items with arbitrarily large $|L|$. In this paper, we use LtR and personalized LtR interchangeably unless otherwise stated. Our focus is *personalized pairwise LtR* in implicit feedback scenarios.

Personalized pairwise LtR methods can directly learn from a user's preference information between a pair of items. This differs from the typical pointwise LtR approaches to implicit feedback where all user interactions are given a positive label, and unknown interactions are either ignored, or given a negative class label which may arguably lead to an undesired learning model [15]. Pairwise LtR methods can also be more efficient in some problems compared to listwise LtR since they work on item pairs instead of arbitrarily large lists. Therefore, given the usefulness of pairwise LtR for implicit feedback scenarios, our motivation in this paper is to further improve its scalability to large feedback datasets and streams.

The idea of shared memory lock-free parallel SGD is motivated for different machine learning methods, and applied to matrix completion from sparse feedback data in [13, 12]. These approaches are especially interesting for multi-core CPU or GPU processing, and they can be superior to approaches like MapReduce [9] for memory-based numerical computation.

In this paper, we investigate a shared memory lock-free parallel SGD scheme for personalized pairwise LtR to improve its scalability in online and stream processing settings. We first show that such a parallel online algorithm is directly applicable to pairwise LtR as in pointwise LtR given in [13], and propose the parallel P-BPR algorithm. We then show that a few modifications to this parallel online algorithm help solve several problems arising from stream processing of implicit user feedback, and propose the PS-BPR algorithm.

We structure our paper as follows: In Section 2, we analyze parallel personalized pairwise LtR and propose the two algorithms. In Section 3, comparative experimental results are provided for the ranking ability of the algorithms and the speedup patterns. We mention related work in Section 4, and our conclusions in Section 5.

2 Methods

We begin this section with a subsection providing some notation and background on personalized pairwise LtR. Then, in the following two subsections, we propose and discuss our two algorithms.

2.1 Personalized pairwise LtR basics

Given a set of users, U, and a set of items, I, personalized pairwise LtR methods commonly learn from triples (u, i, j), where $u \in U$ is the context, and $i, j \in I$. Furthermore, $i \in I_u^+$, $j \in I \setminus I_u^+$, where I_u^+ denotes the set of items interacted by user u in a positive sense. We note that i and j may have additional semantics in enriched contexts. A triple implies a partial personalized ranking such that for a given user u, the ranking $r^i \prec_u r^j$ holds. In other words, user u prefers item i over item j. The ultimate goal is to discover full or topmost personalized rankings from available partial rankings.

In this paper, we stick to a seminal pairwise LtR approach, Bayesian personalized ranking (BPR) [15, 14] using matrix factorization. Under independence assumptions, this approach obtains the following estimate of model parameters:

$$\begin{aligned} p\left(\mathbf{P}, \mathbf{Q} | \prec_u\right) &\propto p\left(\prec_u | \mathbf{P}, \mathbf{Q}\right) p\left(\mathbf{P}, \mathbf{Q}\right), \\ p\left(\mathbf{P}, \mathbf{Q} | \prec_u\right) &\propto \prod_{(u,i,j) \in \mathcal{D}} \sigma\left(x_{uij}\right) p\left(\mathbf{P}, \mathbf{Q}\right), \end{aligned} \tag{1}$$

where $\prec_u$ represents all available partial rankings in a given dataset $\mathcal{D}$, and $\sigma\left(.\right)$ is the logistic function. x_{uij} is estimated using low-rank matrix factorization, in which case we have two low-rank matrices $\mathbf{P} \in \mathbb{R}^{|U| \times f}$ and $\mathbf{Q} \in \mathbb{R}^{|I| \times f}$ where typically $f << |U|, |I|$. Then, it can be chosen that $x_{uij} = \mathbf{p}_u^\top \left(\mathbf{q}_i - \mathbf{q}_j\right)$ using factorized representations of the user and items, that is, $\mathbf{p}_u$ and $\mathbf{q}_k$ are column vectors of

[1] Department of computer engineering, Boğaziçi University, email: murat.yagci@boun.edu.tr
[2] Department of computer engineering, Bahçeşehir University, email: tevfik.aytekin@bahcesehir.edu.tr
[3] Clear Group, email: huerol.tueren@clear-it.de
[4] Department of computer engineering, Boğaziçi University, email: gurgen@boun.edu.tr

$\mathbf{P}$ and $\mathbf{Q}$, respectively. In other words, the model favors a higher dot product score with the preferred item compared to the dot product score with the less preferred item. By using negative log likelihood, this formulation can be more conveniently written as,

$$\mathcal{F}(\mathbf{P}, \mathbf{Q}) = -\sum_{(u,i,j)\in\mathcal{D}} \ln \sigma(x_{uij}) + \lambda_{\mathbf{P}}||\mathbf{P}||^2 + \lambda_{\mathbf{Q}}||\mathbf{Q}||^2, \quad (2)$$

where $\lambda_.$ is a regularization parameter. This is differentiable with respect to model parameters and can be optimized with SGD using a suitable learning rate, γ. The procedure is summarized in Algorithm 1. Note that the algorithm can perform multiple training epochs over data, but there is no need for shuffling prior to each epoch since sampling with replacement is used.

Algorithm 1: BPR learning algorithm using matrix factorization

```
repeat
    Sample a triple (u, i, j) from D ;
    foreach θ ∈ (p_u, q_i, q_j) do
        θ = θ + γ ∂/∂θ {ln σ(x_uij) − λ_θ ||θ||^2} ;
until convergence;
```

2.2 Parallelization of pairwise LtR

For shared memory lock-free parallelization of matrix completion, one idea is block partitioning which partitions a user-item interaction matrix into multiple sets of non-overlapping parts [12, 3]. Then, each processing unit updates its own part in every set which enables a form of parallel processing without using locks. This approach has been originally applied to personalized pointwise LtR scenarios, where updates are based on a single item interaction. In the case of pairwise LtR, a block partitioning solution can be formulated as in Figure 1 regarding the (u, i, j) updates. A possible drawback of this solution is that the partitions in each set need to have a balanced number of training tuples in order to fully benefit from parallelization. Furthermore, too many sets of partitions add additional costs due to synchronization before each set is processed. Since the training data has to be permuted suitably prior to optimization, a further complication is that it is not straightforward to apply this approach to dynamic and stream processing environments, for example, where context or items change frequently, or online sampling is performed.

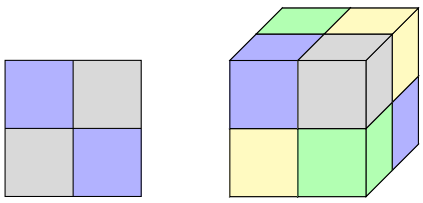
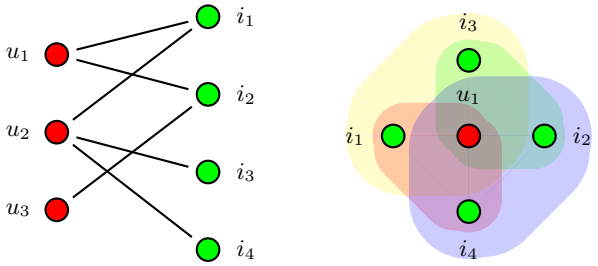


Figure 1: (Left) With two processing units, parallel pointwise LtR can be parallelized with such a partitioning of the user-item interaction matrix. The processing units first compute the tuples belonging to partitions of one color and then those belonging to the other. (Right) In pairwise LtR, the second item can be imagined on a third dimension. More partitions are needed to guarantee mutually exclusive updates to users and items.

Another approach is not to apply any partitioning [13], that is, parallel processes can access and write any portion of the shared memory at any time and in a lock-free fashion. Nevertheless, this extreme approach still provides some theoretical guarantees for convergence when the optimization is sparse and the optimization function can be defined as summations. These requirements are typical of online matrix factorization for personalized LtR. In the case of BPR, the summation has the following sparse form:

$$\sum_{(u,i,j)\in E} \{-\ln \sigma(x_{uij}) + \lambda'_{\mathbf{P}}||\mathbf{p}_u||^2 + \lambda'_{\mathbf{Q}}||\mathbf{q}_i||^2 + \lambda''_{\mathbf{Q}}||\mathbf{q}_j||^2\}. \quad (3)$$

This summation induces a hypergraph, $G = (V, E)$. However, unlike a bipartite graph for pointwise LtR as specified in [13], we now have a hypergraph which reflects personalized pairwise item relationships. We illustrate this new hypergraph for pairwise LtR in comparison to the bipartite graph for pointwise LtR in Figure 2.



(a) (b)

Figure 2: Example hypergraphs showing user-item relationships for pointwise and pairwise LtR respectively. For clarity, the latter is shown for a single user only, where 3-vertex hyperedges capture all possible pairwise item relationships for u_1. Here, $i_1, i_2 \in I^+_{u_1}$ and $i_3, i_4 \in I \setminus I^+_{u_1}$.

Both $|E|$ and $|V|$ are typically very large with the former being especially large in the case of pairwise LtR. However, as seen in (3), each summation acts on a single $e \in E$, that is, a very small subset of V, which suggests intuitively that parallel lock-free updates without any partitioning are viable. Following the theoretical analysis in [13], convergence of the optimization is affected by the following statics of the hypergraph, G:

$$\begin{aligned} \Omega &= \max_{e\in E} |e|, \\ \Delta &= \frac{\max_{1\le v\le |V|} |\{e \in E : v \in e\}|}{|E|}, \\ \rho &= \frac{\max_{e\in E} |\{e' \in E : e' \cap e \neq \emptyset\}|}{|E|}, \end{aligned} \quad (4)$$

where Δ and ρ are measures of vertex regularity and hypergraph sparsity, respectively. In pairwise LtR, $\Omega = 3$, since each e is made up of an (u, i, j) triple. Assume every user makes m interactions, then $|E| = |U| \times m \times (|I| - m)$ triples. Since we have (u, i, j) triples, there can be user and item vertices in the hypergraph. Assume an item is interacted with at most n users. Then, it can be that $\Delta = (|I| - m) \times m/|E|$ regarding user vertices, or $\Delta = [(|I| - m) \times n + m \times (|U| - n)]/|E|$ regarding item vertices. As expected, we see that item vertices determine Δ in many real life datasets. However, it is not straightforward to make general assumptions about m and n in the worst case. For example, in case $m << |I|$, $n << |U|$, and $m \approx n$, $\Delta \approx 1/|U| + 1/|I|$ which is often close to $\Delta \approx \log(|U|)/|U|$ of pointwise LtR under similar assumptions. Since $\Omega = 3$, $\rho \le 3\Delta$. In real datasets, the shared memory lock-free approach without any partitioning is effective even

when the values of Δ and ρ are relatively high, as shown experimentally in [13] for different machine learning problems including matrix completion in a pointwise LtR setting. We observe in our experiments that in the personalized LtR problem setting, a possible reason behind this situation is that user and item vertex degrees often follow a power law distribution, that is, Δ (maximum normalized vertex degree) can be significantly higher than the values in the modal interval of the normalized degree distribution. In Section 3, we compute all these statistics for various datasets and show that the statistics of pairwise LtR are quite similar to those of pointwise LtR.

Therefore, we propose to parallelize shared memory online processing of BPR without any partitioning as given in Algorithm 2. We call this algorithm P-BPR. As an illustrative example, if we assume π processing units and 2π total epochs of the learning algorithm, then we see that every processing unit will run 2 epochs instead of a single processing unit running 2π epochs.

Algorithm 2: Parallel online BPR algorithm (P-BPR)

```
// Perform following loop in every parallel
   processing unit
foreach training epoch do
    foreach iteration do
        Sample a triple (u, i, j) ;
        Read current state of (p_u, q_i, q_j) and compute
        gradients w.r.t. (2) ;
        foreach θ ∈ (p_u, q_i, q_j) do
            Update θ with its gradient ;
    Synchronize processing unit if learning rate, γ, changes ;
```

2.3 Learning from streaming data

Streaming algorithms [9] are useful when trying to learn from data in a single pass without storing the entire dataset, and using low cost summary structures. Such algorithms can be used for large-scale learning problems as well as real-time learning. Before our discussion, we list below some general problems for LtR with low-rank matrix factorization and streaming data:

1. Single pass often yields a poor factorization due to slow convergence with small learning rates and unguaranteed convergence when higher learning rates are used.
2. Low-rank factorization model offers lower cost data structures holding latent factor representations. But the whole dataset should not be kept in main memory either, and incremental model updates should be done.
3. Fine tuning model parameters such as learning rate is even harder in the streaming setting where there may be concept drift. Adaptivity to non-stationary processes is needed. A forgetting mechanism is useful.
4. Apart from learning optimization parameters, the top-N prediction step in personalization is also expensive in matrix factorization. Learning in a single pass itself may not be enough for real-time prediction.
5. Parallel computing architectures should be exploited for streaming algorithms regarding today's web-scale systems.

Algorithm 2 actually has a solution for the first problem with parallel updates. Assume we have π processing units. In the general case, a processing unit can perform $\lceil \frac{\#\text{total epochs}}{\pi} \rceil$ training epochs. However, when we choose to perform one epoch in each processing unit, we still get π epochs in a single pass over data amounting to π sequential epochs. Therefore, even if we choose small learning rates, since we perform π parallel updates, convergence is expected to be faster.

For the second problem, reading data one by one from the stream and doing immediate updates in parallel is not a good idea even if we can simulate some randomness by online sampling with replacement, for example, using Oza's method [11]. This is due to parallel updates of the same tuple. To this end, we propose a scheme described in Algorithm 3 by modifying Algorithm 2. Instead of a possible reservoir sampling approach as in [1], we perform non-overlapping sliding windows over data. When a sliding window buffer is full, parallel processing units perform their updates and then wait for the next buffer. Algorithm 3 has two benefits: First is to keep a fixed size buffer for the data and second is temporal locality due to incremental updates. These benefits are further discussed in Section 3. On the other hand, if the stream is massive, there needs to be some stricter synchronization mechanism between the producer and the consumers. In this case, one approach can be to use a subsample size (T) smaller than the buffer size.

Algorithm 3: Parallel BPR algorithm for streams (PS-BPR)

```
// Perform following loop in stream
   producer
1  foreach tuple (u, i) from stream do
2      Insert (u, i) into a sliding window buffer ω in shared
       memory ;
3      if ω is full then
4          Initiate consumers with ω ;
5          Synchronize consumers ;              // Optional
6          Initiate new buffer ω ;

   // Perform the following in every parallel
      processing unit (consumers)
7  τ = 0 ;
8  while τ < min (length (ω) , T) do
9      Sample a tuple (u, i) from ω uniformly randomly ;
10     Sample j ∈ I \ I_u^+ uniformly randomly;
11     Read current state of (p_u, q_i, q_j) and compute gradients
       w.r.t. (2) ;
12     foreach θ ∈ (p_u, q_i, q_j) do
13         Update θ with its gradient ;
14     τ = τ + 1 ;
```

For the third problem, Algorithm 3 provides several possible improvements: First, due to parallel updates, we can choose a relatively small learning rate initially without the need for adjusting it later. We note that opting for adjusting the learning rate requires some sort of synchronization among processes (see last line in Algorithm 2), which can be optionally implemented in Algorithm 3 (see line 5). Second, using sliding windows over data enables at certain points to forget the trained model so far and start to train a second model to adapt to concept drift.

Algorithm 3 is also an approach towards the fifth problem since the streaming algorithm is coupled with a parallelization scheme which further improves its efficiency. We do not explicitly deal with the fourth problem in this paper, but we note that the top-N prediction step can also be parallelized.

3 Experiments

We begin this section by describing the data and our experimental setup. Then, we compare statistics of dataset graphs for pointwise and pairwise LtR. We conclude by presenting detailed comparative experimental results for P-BPR and PS-BPR algorithms in terms of ranking ability and speedup.

3.1 Datasets and evaluation

We use MovieLens 1M (ML1M), MovieLens[5] 10M (ML10M) [4], and the latest collection of MovieTweetings[6] (MTWT) [2] datasets in our experiments. Basic properties of the datasets are given in Table 1. Raw datasets consist of (u, i, t) tuples where t is a timestamp. Since we are interested in implicit feedback, we are only concerned with whether or not a user interacts with a specific item. Therefore, we do not use the available rating information in training our models. We sort the datasets in time order and use the first 80% of time-ordered datasets for training purposes and the remaining 20% for testing.

Table 1: Basic properties of datasets

	# Users	# Items	# Interactions
ML1M	6,040	3,900	1,000,209
ML10M	71,567	10,681	10,000,054
MTWT	44,046	25,529	522,233

We measure ranking ability of the algorithms using mean average precision, MAP@N [14, 16]. For each user in the test set, we predict a top-N list using the trained model with a cut-off value $N = 1000$. This ordered list contains N items with topmost ranking scores, which the user has not yet interacted with, and it is compared to user's interactions in the test set. Average precision, AP@N, is an approximation of the area under precision-recall curve by summing over the product of precision and change in recall at values up to N. MAP@N is the average over all query users. Besides ranking evaluation, we illustrate speedup of the algorithms graphically using execution times.

We use a virtual cloud machine having a multi-core CPU with 12 physical cores and enough RAM. We rely on the Linux operating system and Python's multiprocessing to exploit these cores. We monitor correct usage using diagnostic tools. We also use shared memory numpy structures for matrix operations. We leave the atomicity of operations to what is available in the CPU.

3.2 Statistics of dataset graphs

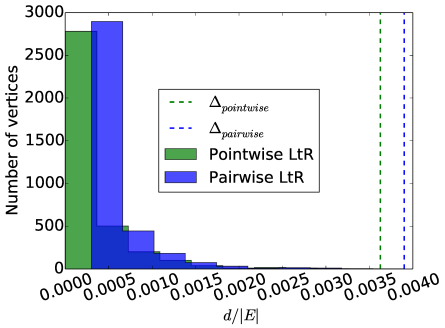
We enumerate the (u, i, j) triples in the training sets and see that both Δ and ρ are on the order observed in real life datasets for pointwise LtR in [13]. We note that such an enumeration is never necessary during the course of P-BPR or PS-BPR, but a random sampling scheme is performed instead. We provide comparative statistics in Table 2 which indicate similar convergence properties for pointwise and pairwise LtR.

We also investigate the vertex degree distributions of the dataset graphs and compare them with the Δ values in Figure 3. We observe that although the Δ values define an upper bound for normalized vertex degree distributions, the modal intervals of the distributions correspond to much lower values which further support parallel methods like P-BPR and PS-BPR.

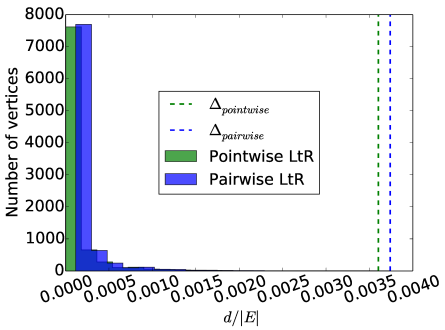
[5] http://grouplens.org/datasets/
[6] https://github.com/sidooms/MovieTweetings

Table 2: Statistics of dataset graphs for LtR

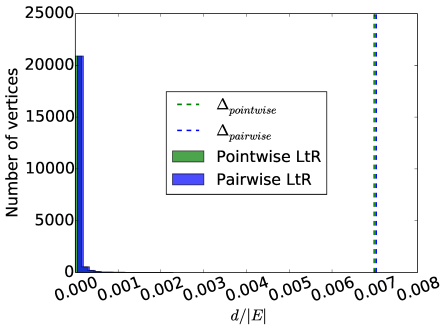
	Δ	ρ
ML1M - Pointwise LtR	0.0036	$\leq 2\Delta$
ML1M - Pairwise LtR	0.0039	$\leq 3\Delta$
ML10M - Pointwise LtR	0.0036	$\leq 2\Delta$
ML10M - Pairwise LtR	0.0037	$\leq 3\Delta$
MTWT - Pointwise LtR	0.0070	$\leq 2\Delta$
MTWT - Pairwise LtR	0.0070	$\leq 3\Delta$



(a) ML1M



(b) ML10M

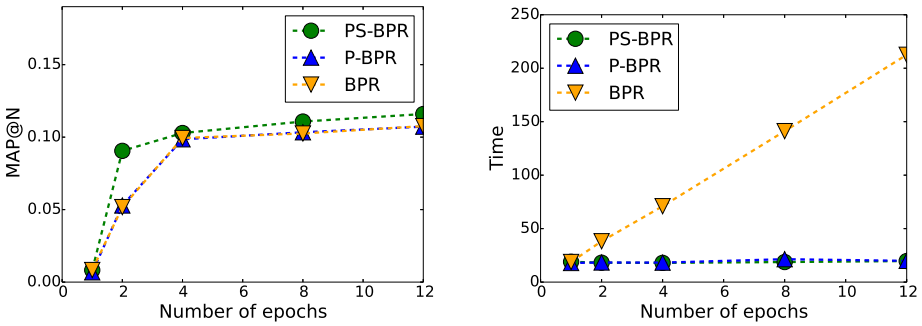


(c) MTWT

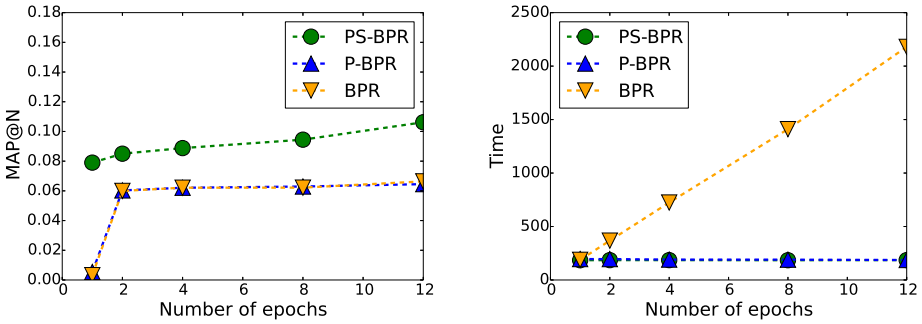
Figure 3: Vertex degree (d) distributions in comparison to Δ values for different datasets

3.3 P-BPR

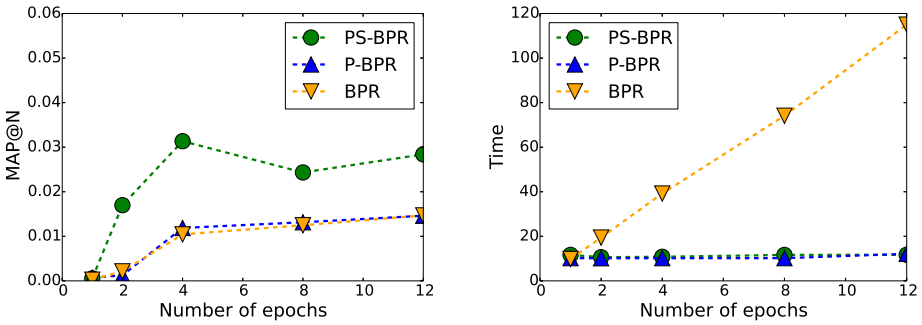
Experimental results for P-BPR are obtained by performing a single epoch in each processing unit, amounting to π parallel epochs. We test up to 12 epochs similar to the experimental setup in [13], and based on the observation [5] that 10 epochs is often a good trade-off between accuracy and training time in industrial systems. The results are illustrated in Figure 4. The important comparison



(a) ML1M



(b) ML10M



(c) MTWT

Figure 4: Comparison of ranking ability and speedup for BPR, P-BPR, and PS-BPR

here is to BPR which performs π epochs sequentially. The learning rate, γ, equals 0.005, 0.010, and 0.015 for ML1M, ML10M, and MTWT, respectively. The following parameters are the same for all datasets and given here for reproducibility of results: Regularization parameters are found by cross-validation using a small search space around $\lambda'_{\mathbf{P}} = \lambda'_{\mathbf{Q}} = 0.0025$, $\lambda''_{\mathbf{Q}} = 0.00025$. Number of latent factors, $f = 10$. Initial values of matrices $\mathbf{P}$ and $\mathbf{Q}$ are sampled from $\mathcal{N}(0, 0.01)$.

We observe in all datasets that the difference between ranking ability of BPR and P-BPR is not statistically significant which shows the effectiveness of the shared memory lock-free parallelization. On the other hand, the speedup patterns show that P-BPR scales very well irrespective of the number of total epochs whereas, as expected, sequential BPR scales linearly. These results suggest that P-BPR can converge very fast scaling to the resources at hand and without loss of ranking ability.

3.4 PS-BPR

Experimental results for PS-BPR are obtained by performing π parallel updates as soon as a sliding window buffer becomes full. The results are illustrated again in Figure 4. The learning rates and other parameters are the same as those of BPR and P-BPR. $T = length\,(\omega)$, which is 1/10th of the training set for MTWT and 1/20th of it for the others.

The MAP@N pattern of PS-BPR is interestingly much better than BPR and P-BPR, especially with ML10M and MTWT. The convergence is faster and the ranking ability is higher. We observe that following the time order in updates is useful. Moreover, exploiting temporal locality in a sliding window fashion results in a more useful sampling of j for each (u, i) since preferences evolve over time. In terms of space complexity, PS-BPR achieves $O\,(length\,(\omega))$ for holding data instead of holding the full dataset. On the other hand, the speedup pattern shows that PS-BPR can achieve execution times

comparable to P-BPR which means that it also scales quite well. It should be noted that this speedup comparison is only in terms of training times, and in practice streaming approach also has the additional benefit of learning incrementally from smaller sliding windows rather than awaiting all data to gather for batch updates.

4 Related work

Shared memory lock-free parallel matrix completion without any data partitioning scheme is motivated in [13] and some preliminary results are shown for personalized pointwise LtR. Our work extends the idea to personalized pairwise LtR, and streaming algorithms as well as online algorithms. Furthermore, our experimental setting is focused on the ranking ability of algorithms rather than reducing training error.

BPR [15, 14] algorithm is widely used for pairwise LtR. Our algorithms improve efficiency of BPR without loss of ranking ability. BPR uses AUC as the optimization criterion with a sigmoid approximation. In [1], authors prefer hinge loss for pairwise LtR. Their algorithm is presented as a streaming algorithm which uses reservoir sampling to maintain a representative sample of stream rather than our sliding window approach. However, they do not apply any parallellization.

5 Conclusions and future work

Personalized pairwise LtR is a powerful approach for learning from relevance feedback datasets. We aim to improve its scalability and adaptivity using a shared memory lock-free parallel SGD scheme. In this direction, we propose two algorithms: P-BPR as an online algorithm and PS-BPR as a streaming algorithm. We show that the usefulness of these new algorithms is due to the graph theoretical similarities between pairwise and the formerly analyzed pointwise LtR, for which the same shared memory lock-free SGD scheme has good convergence properties. Our experimental results support this analysis further with respect to ranking ability and speedup of the new algorithms compared to their sequential counterpart. Moreover, using time information and temporal locality, we observe additional improvements in ranking ability and convergence speed with time-ordered datasets. P-BPR and PS-BPR exploit modern multi-core and shared memory architectures, and they are easy to implement which can make them desirable in web-scale applications.

We note the following as future work: GPU-based parallel processing can be exploited for experimental limits of parallelization. Furthermore, the mutual effect of biased sampling schemes [14] and parallelization needs further investigation.

REFERENCES

[1] Ernesto Diaz-Aviles, Lucas Drumond, Lars Schmidt-Thieme, and Wolfgang Nejdl, 'Real-time top-n recommendation in social streams', in *Proceedings of the 6th ACM Conference on Recommender Systems*, RecSys '12, pp. 59–66, (2012).

[2] Simon Dooms, Toon De Pessemier, and Luc Martens, 'Movietweetings: a movie rating dataset collected from twitter', in *Workshop on Crowdsourcing and Human Computation for Recommender Systems, CrowdRec at RecSys '13*, (2013).

[3] Rainer Gemulla, Erik Nijkamp, Peter J. Haas, and Yannis Sismanis, 'Large-scale matrix factorization with distributed stochastic gradient descent', in *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '11, pp. 69–77, (2011).

[4] F. Maxwell Harper and Joseph A. Konstan, 'The movielens datasets: History and context', *ACM Transactions on Interactive Intelligent Systems*, **5**(4), 19:1–19:19, (2015).

[5] Balazs Hidasi and Domonkos Tikk, 'General factorization framework for context-aware recommendations', *Data Mining and Knowledge Discovery*, **30**(2), 342–371, (2016).

[6] Yifan Hu, Yehuda Koren, and Chris Volinsky, 'Collaborative filtering for implicit feedback datasets', in *Proceedings of the 8th IEEE International Conference on Data Mining*, ICDM '08, pp. 263–272, (2008).

[7] Alexandros Karatzoglou, Linas Baltrunas, and Yue Shi, 'Learning to rank for recommender systems', in *Proceedings of the 7th ACM Conference on Recommender Systems*, RecSys '13, pp. 493–494, (2013).

[8] Yehuda Koren, Robert Bell, and Chris Volinsky, 'Matrix factorization techniques for recommender systems', *Computer*, **42**(8), 30–37, (2009).

[9] Jure Leskovec, Anand Rajaraman, and Jeffrey David Ullman, *Mining of massive datasets*, Cambridge University Press, 2014.

[10] Tie-Yan Liu, *Learning to Rank for Information Retrieval.*, Springer, 2011.

[11] N.C. Oza, 'Online bagging and boosting', in *Proceedings of the IEEE Conference on Systems, Man and Cybernetics*, volume 3, pp. 2340–2345, (2005).

[12] Benjamin Recht and Christopher Ré, 'Parallel stochastic gradient algorithms for large-scale matrix completion', *Mathematical Programming Computation*, **5**(2), 201–226, (2013).

[13] Benjamin Recht, Christopher Re, Stephen Wright, and Feng Niu, 'Hogwild: A lock-free approach to parallelizing stochastic gradient descent', in *Advances in Neural Information Processing Systems*, 693–701, (2011).

[14] Steffen Rendle and Christoph Freudenthaler, 'Improving pairwise learning for item recommendation from implicit feedback', in *Proceedings of the 7th ACM International Conference on Web Search and Data Mining*, WSDM '14, pp. 273–282, (2014).

[15] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme, 'Bpr: Bayesian personalized ranking from implicit feedback', in *Proceedings of the 25th Conference on Uncertainty in Artificial Intelligence*, UAI '09, pp. 452–461, (2009).

[16] *Recommender Systems Handbook*, eds., Francesco Ricci, Lior Rokach, and Bracha Shapira, Springer, 2015.

[17] Yue Shi, Alexandros Karatzoglou, Linas Baltrunas, Martha Larson, Nuria Oliver, and Alan Hanjalic, 'Climf: Learning to maximize reciprocal rank with collaborative less-is-more filtering', in *Proceedings of the 6th ACM Conference on Recommender Systems*, RecSys '12, pp. 139–146, (2012).

On the Identifiability of Models in Multi-Criteria Preference Learning

Christophe Labreuche[1], Eyke Hüllermeier[2], Peter Vojtas[3], Ali Fallah Tehrani[4]

Abstract. In preference learning, the identifiability of models in the sense of the uniqueness of their parameterization is a desirable property, not only from an algorithmic point of view but also with regard to the interpretability of the model. In multi-criteria preference learning, models are used that aggregate several local utility degrees (pertaining to individual criteria) into a global evaluation of a choice alternative. Due to the interplay between local and global utility functions, the identifiability of such models is no longer obvious. In this paper, we analyze the question of identifiability for a class of models in which the (discrete) Choquet integral is used as an aggregation function.

1 Introduction

Multi-criteria decision aid (MCDA) aims at helping a decision maker to rank or sort choice alternatives on the basis of their values on multiple criteria [8, 7]. To this end, MCDA has developed a wide variety of decision models, most of which aggregate the evaluations on individual criteria into an overall assessment of an alternative. An inportant example of a model of that kind is the (discrete) Choquet integral [1]. The specification of models in MCDA is typically accomplished in the course of an interactive process, in which a decision analyst seeks to elicit the decision maker's preferences by asking informative questions [11].

In contrast to such kind of human-centric *construction* of preference models, preference learning (PL) is geared towards the automated *induction* of models from large amounts of data [2]. Moreover, there are other notable differences between PL and MCDA. For example, while MCDA mostly focuses on a single decision maker, a model in PL typically refers to an entire polulation of individuals. Moreover, while user feedback in MCDA is assumed to be consistent, or inconsistencies can be repaired by the decision maker, PL generally allows the observed data to be noisy. Last but not least, PL puts very much emphasis on generalization performance and predictive accurary.

In spite of these differences, or rather because of them, PL and MCDA can nicely complement each other, and indeed, quite a number of methods in the intersection of both fields have been developed in recent years. An important example is what we call *multi-criteria preference learning* (MCPL): the combination of *models* from MCDA with *algorithms* from machine learning or, stated differently, the automated induction of MCDA models from (possibly noisy) data. In contrast to many other machine learning approaches, MCPL produces models that are interpretable and meaningful from of decision making point of view. As a concrete examples, let us mention recent work on methods for learning the majority rule model [13], the non-compensatory sorting model [14], or the Choquet integral [16, 5, 17].

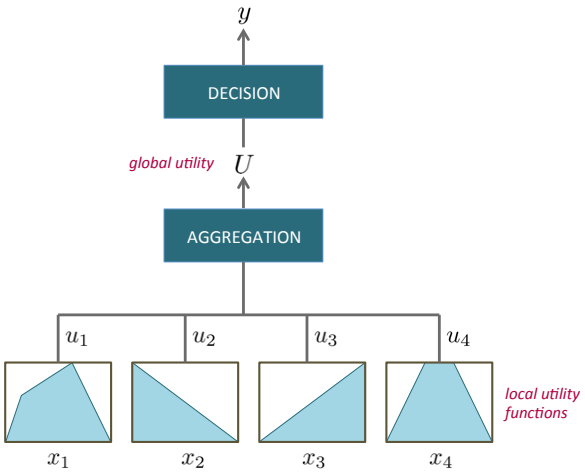


Figure 1. General structure of an MCPL model.

Figure 1 shows the general structure of an MCPL model: Given a choice alternative described in terms of an attribute vector $\boldsymbol{x} = (x_1, \ldots, x_m)$, each attribute x_i is first evaluated by means of a *local utility function*, and thereby turned into a *local utility degree* $u_i = u_i(x_i) \in \mathbb{R}$. In a second step, the local utility degrees $u_1, \ldots, u_m$ are aggregated into a *global utility* $U = U(u_1, \ldots, u_m)$. Finally, a decision y (for example, assigning a rank to an alternative) is taken based on this utility; considering the noisy nature of the observed data, the dependency between U and y is modeled in terms of a probabilistic model P. Eventually, we thus obtain a model that can be written (in a somewhat simplified form) as follows:

$$y = P(U) = P\Big(U\big(u_1(x_1), \ldots, u_m(x_m)\big)\Big) = G_\theta(x_1, \ldots, x_m)$$

[1] Thales Research & Technology, 91767 Palaiseau cedex, France
[2] Dept. of Computer Science, Paderborn University, Germany
[3] Charles University, Prague, Czech Republic
[4] Technische Hochschule Deggendorf, Germany

The decision y is a parameterized function of $\boldsymbol{x}$, where the vector θ comprises the parameters of the local utilty functions u_i, the aggregation function U, and the probabilistic model P. The problem we address in this paper is the question of the *identifiability* of the model: Is the dependency between $\boldsymbol{x}$ and y uniquely defined by the parameter θ, or is it possible that $G_\theta \equiv G_\vartheta$ for $\theta \neq \vartheta$?

The answer to this question is important for various reasons. Firstly, unique solutions often facilitate the learning and render optimization algorithms (such as gradient descent) more efficient and stable. Secondly, and perhaps more importantly, uniqueness of the model is a prerequisite for a meaningful interpretation, which, as already said, is a major advantage of MCDA models.

At the same time, the answer to the question of identifiability is in general non-trivial. Indeed, one can easily imagine cases in which a modification in the local utility functions u_i can be "annulled" by a corresponding modification of the aggregation U, so that both modifications (parameter changes) eventually compensate each other. For an algorithm that seeks to learn $u_1, \ldots, u_m$ and U simultaneously, this could be problematic.

Building on our own previous work [16, 19], we study the question of identifiability for models in which the probabilistic model P is a generalization of logistic regression, and the aggregation U is realized by the Choquet integral. An overview of the three parts of the model, i.e., the decision model P, the aggregation U, and the local utility functions u_i, is given in Sections 2, 3, and 4, respectively. Our results on identifiability are then presented in Section 5, prior to concluding the paper in Section 6.

2 Probabilistic Decision Model

Let us consider the "upper" part of the MCPL model first, i.e., the question of how y depends on the utility U via a model P. In [16], a model is introduced that only distinguishes between two decisions, namely $y = 1$ ("good alternative") and $y = 0$ ("bad alternative"). Moreover, the dependency between (global) utility and decision is captured by an assumption that is also made in standard logistic regression, namely that the log-odds ratio is an affine function of the utility U:

$$\log\left(\frac{\mathbf{P}(y=1\,|\,\boldsymbol{x})}{\mathbf{P}(y=0\,|\,\boldsymbol{x})}\right) = \gamma\, U\,(u_1,\ldots,u_m) + \beta \ , \qquad (1)$$

where $\mathbf{P}(y = 1\,|\,\boldsymbol{x})$ is the probability that an alternative with feature description $\boldsymbol{x}$ is evaluated as "good" and $\mathbf{P}(y = 0\,|\,\boldsymbol{x}) = 1 - \mathbf{P}(y = 1\,|\,\boldsymbol{x})$ the complementary probability of a "bad" evaluation. Setting $P = \mathbf{P}(y = 1\,|\,\boldsymbol{x})$, the assumption (1) is equivalent to the probabilistic response model

$$P = \frac{1}{1 + \exp\big(-\gamma\, U(u_1,\ldots,u_m) - \beta\big)} \ . \qquad (2)$$

According to (2), the larger the (global) utility U, the larger the probability to observe $y = 1$. The concrete form in which the probability P depends on U is controlled by the scaling parameter $\gamma > 0$ and the bias term β.

3 Global Utility Function

In this section, we look at the "middle" part of the MCPL model, that is, the way in this the global utility U depends on the local utilities $u_1, \ldots, u_m$ via an aggregation function.

The utility functions map the attribute vector $\boldsymbol{x} = (x_1, \ldots, x_m)$ onto an overall utility degree. In general, the attributes can take discrete (e.g. labels) or continuous values. In the former case, in order to account for the monotonicity conditions, it is more convenient to represent x_i as an integer value (where the integer is the rank order of the label according to their relative preference) rather than a label. We assume thus in all cases that $x_i \in \mathbb{R}$.

3.1 Logistic and Choquistic Regression

The simplest overall utility function U is a weighted sum

$$U(u_1,\ldots,u_m) = \boldsymbol{\omega}^\top \boldsymbol{u} = \sum_{i=1}^{m} \omega_i\, u_i \,, \qquad (3)$$

where the parameters are the weights $\boldsymbol{\omega} = (\omega_1, \ldots, \omega_m)$, which satisfy $\omega_i \geq 0$ and $\omega_1 + \ldots + \omega_m = 1$. The application of (1) with this model yields standard logistic regression (LR).

The Choquet integral [1] has been used in MCDA as a generalization of the weighted sum, mainly to relax the assumption that criteria do not interact with each other [3]. This aggregation function is based on the concept of a *capacity* [1], also called *fuzzy measure* [15]. Let $M = \{1, \ldots, m\}$ be the set of criteria. A capacity is a set function $v : 2^M \to \mathbb{R}$ such that

$$v(\emptyset) = 0, \quad v(M) = 1 \text{ (normalization)} \qquad (4)$$

$$v(A) \leq v(B) \text{ whenever } A \subseteq B \subseteq M \text{ (monotonicity)} \qquad (5)$$

Roughly speaking, $v(A)$ is the importance of the set of criteria A. The Choquet integral prescribes how to aggregate utilities $u_1, \ldots, u_m$ with respect to a non-additive measure v:

$$C_v(u_1,\ldots,u_m) = \sum_{i=1}^{m} (u_{\tau(i)} - u_{\tau(i-1)})\; v(\{\tau(i),\ldots,\tau(n)\})$$

where τ is a permutation on M such that $0 =: u_{\tau(0)} \leq u_{\tau(1)} \leq \cdots \leq u_{\tau(n)}$. We refer the reader to [3, 4] for more detailed exposition of the use of the Choquet integral in MCDA.

The model based on the Choquet integral takes the following form:

$$U(u_1,\ldots,u_m) = C_v\big(u_1,\ldots,u_m\big) \qquad (6)$$

Thus, the model is parameterized by the capacity v. In [16], the instantiation of (1) with this model is called Choquistic Regression (CR). As a side remark, we note that LR is obtained as a special case of CR, namely if the capacity reduces to an additive measure.

3.2 Choquistic Utilitaristic Regression

Both LR and CR as considered in [16] assume the local utility functions u_i to be given. More specifically, attribute values x_i are turned into utilities u_i through simple normalization or standardization of the data. Note that, since (3) and (6) are necessarily monotone increasing in each local utility, the original attributes are then assumed to be criteria, i.e., "higher-is-better" attributes.

The importance of learning the local utility functions, jointly with the aggregation, has been shown in MCDA [12].

Therefore, we are now looking at "utilitaristic" generalizations of LR and CR, in which the local utility functions are parameterized; concrete examples of such functions will be given later on in Section 4.

First, we consider a utilitaristic version of LR. Note that, provided the local utilities u_i are normalized to the unit interval, the same holds true for the global utilities (3) and (6). From a practical side, it is sometimes more convenient to express the utility U as a plain sum

$$U(u_1, \ldots, u_m) = \sum_{i \in M} u_i \tag{7}$$

with possibly non-normalized local utilities $u_i = u_i(x_i)$. Obviously, (3) is recovered as a special case, simply by absorbing the coefficients ω_i in u_i. The application of (1) with this model, in which the u_i are parameterized functions amenable to data-driven adaptation, is called Utilitaristic Regression (UR) [19]. Note that, for this approach, we can set $\gamma = 1$ in (1) without loss of generality.

The last model is a utilitaristic extension of CR, which combines the learning of local utility functions with the ability of CR to represent interaction between features. This gives the following model

$$U(u_1, \ldots, u_m) = C_v\big(u_1, \ldots, u_m\big), \tag{8}$$

i.e., the same as (6), except that the local utilities $u_i = u_i(x_i)$ are not predefined but parameterized functions (see Section 4). The application of (1) with this model is called Choquistic Utilitaristic Regression (CUR) [19].

Note that the use of a Choquet integral requires the utility functions u_i to be commensurate, an assumption that is not needed for the additive utility (7). This is why the local utilites are normalized to the unit interval in (8) but not necessarily in (7).

4 Local Utility Functions

An imporant added value of using models from MCDA is to capture the idea of a *criterion*, that is, to represent a priori information on local preferences restricted to single features (attributes). Several models for the local utility functions can be envisaged, such as piecewise affine functions or linear combination of sigmoids [19]. Nevertheless, some assumptions have to be made.

First of all, the continuity of u_i is desirable in order to avoid a non-stable behavior of the local and hence the global utility. Moreover, in MCDA, the local utility functions are often not fixed uniquely but only up to an affine transformation. In order to get rid of this intrinsic variability, the usual way is to enforce some normalization conditions. The most common assumption is that the local utilities are non-negative and that the smallest possible utility is 0 (a value suggesting that the corresponding criterion is not met at all).

While a maximum utility value is normally not assumed, u_i is nevertheless supposed to be bounded; the corresponding scale is called *bounded unipolar*. For the user, it is much easier to understand models on bounded scales, since otherwise a sufficiently large value on one feature can always compensate any values on the other features.

Finally, we also assume that the local utility functions are not constant, so that the maximum utility is strictly positive. Thus, we can summarize our assumptions as follows:

$$u_i \text{ is continuous on } \mathbb{R} \tag{9}$$

$$\inf_{x_i \in \mathbb{R}} u_i(x_i) = 0 \tag{10}$$

$$0 < \sup_{x_i \in \mathbb{R}} u_i(x_i) < \infty \tag{11}$$

For model CUR, we already mentioned that the local utility functions shall be normalized so that (11) is replaced by the following condition:

$$\sup_{x_i \in \mathbb{R}} u_i(x_i) = 1 \tag{12}$$

4.1 Monotone Utilities

If an attribute is known to be a higher-is-better criterion, relations (9-11) take the following form:

$$u_i \text{ is non-decreasing and continuous} \tag{13}$$

$$\lim_{x_i \to -\infty} u_i(x_i) = 0 \tag{14}$$

$$0 < \lim_{x_i \to +\infty} u_i(x_i) < \infty \tag{15}$$

Relation (12) can be written as

$$\lim_{x_i \to +\infty} u_i(x_i) = 1 \tag{16}$$

We need to use limits when the smallest and greatest values of u_i are not reached at finite elements, that is, for local utility functions with unbounded support; an example is the combinations of sigmoid functions [19]:

$$u_i(x_i) = \sum_{l=1}^{p_i} \frac{r_i^l}{1 + \exp(-\eta_i^l\,(x_i - \lambda_i^l))} \quad , \tag{17}$$

where p_i is the number of sigmoid functions, λ_i^l is the center of the l^{th} sigmoid, $\eta_i^l > 0$ is the slope of the l^{th} sigmoid, and $r_i^l > 0$ controls the relative strengths among the different sigmoid functions. In this model, the smallest and largest values of u_i are reached at $-\infty$ and $+\infty$ respectively. Conditions (14) and (15) are fulfilled as $\lim_{a_i \to +\infty} u_i(a_i) = \sum_{l=1}^{p_i} r_i^l$.

A typical form of a local utility function with bounded support is given by piecewise affine functions, like in the UTA method [6]. Given p_i values $a_i^1 < a_i^2 < \cdots < a_i^{p_i}$, the function u_i is defined as

$$u_i(a_i) = \begin{cases} u_i^1 & \text{if } a_i \le a_i^1 \\ u_i^l + (u_i^{l+1} - u_i^l)\frac{a_i - a_i^l}{a_i^{l+1} - a_i^l} & \\ \qquad\qquad \text{if } a_i^l \le a_i \le a_i^{l+1} & \\ u_i^{p_i} & \text{if } a_i \ge a_i^{p_i} \end{cases} \quad , \tag{18}$$

where $u_i^l = u_i(a_i^l)$ for $l \in \{1, \ldots, p_i\}$.

4.2 Non-monotone Utilities

When monotonicity is not assured, we do not assume any specific shape of the local utility functions. Instead, we would like to cover diverse situations, such as standard monotone

functions (non-decreasing or non-increasing), "single-peaked" functions (such as triangular, trapezoidal, or Gaussian), and V-shaped functions (the inverse of single-peaked functions). Therefore, we only assume (9), (10) and either (11) or (12).

We set

$$\overline{A}_i = \text{argmax}_{x_i \in [-\infty,+\infty]}\, u_i(x_i)$$
$$\underline{A}_i = \text{argmin}_{x_i \in [-\infty,+\infty]}\, u_i(x_i)$$

By (11) or (12), $\overline{A}_i \neq [-\infty,+\infty]$ and $\underline{A}_i \neq [-\infty,+\infty]$.

This model encompasses conventional monotone functions. For a non-decreasing monotone function, $\underline{A}_i$ (resp. $\overline{A}_i$) is of the form $[-\infty, a_i^L]$ (resp. $[a_i^R, +\infty]$). For a non-increasing monotone function, $\underline{A}_i$ (resp. $\overline{A}_i$) is of the form $[a_i^R, +\infty]$ (resp. $[-\infty, a_i^L]$).

Single-peaked functions can also be represented. In this case, $\underline{A}_i$ is of the form $[-\infty, a_i^L] \cup [a_i^R, +\infty]$, and $\overline{A}_i$ is a closed and bounded interval. As a particular case, one finds triangular utility functions (with $a_i^L < a_i^M < a_i^R$):

$$u_i(a_i) = \begin{cases} 0 & \text{if } a_i \leq a_i^L \text{ or } a_i \geq a_i^R \\ \frac{a_i - a_i^L}{a_i^M - a_i^L} & \text{if } a_i^L \leq a_i \leq a_i^M \\ \frac{a_i^R - a_i}{a_i^R - a_i^M} & \text{if } a_i^M \leq a_i \leq a_i^R \end{cases} \tag{19}$$

Another form of single-peaked functions are Gaussians, where $\underline{A}_i = \{-\infty, +\infty\}$.

Inverse single-peaked functions are covered as well. In this case, $\overline{A}_i$ is of the form $[-\infty, a_i^L] \cup [a_i^R, +\infty]$, and $\underline{A}_i$ is a closed and bounded interval.

5 Identifiability

In the previous sections, we discussed representations of the different components of our MCPL model: how local utility function u_i are represented, how the global utility U depends on the local utilities u_i, and how the decision y depends on U. Summarizing these different parts, we end up with a model in which the probability of a positive evaluation of an alternative $\boldsymbol{x}$, denoted by P in (2), is given as follows:

$$G_\theta(\boldsymbol{x}) = \frac{1}{1 + \exp\big(-V_\theta(\boldsymbol{x})\big)} \tag{20}$$

where

$$V_\theta(\boldsymbol{x}) = \gamma\, U(u_1(x_i), \ldots, u_m(x_m)) + \beta \tag{21}$$

The parameterization θ of this model includes

- the parameters of the probabilistic model, γ and β,
- the parameters of the aggregation U, i.e., the coefficients $\boldsymbol{\omega}$ in the case of a linear model or the capacity v in the case of the Choquet integral,
- the parameters of the local utility functions u_i.

As already said, the identifiability of a model refers to the uniqueness of its parameterization, that is, $G_\theta(\boldsymbol{x}) = G_\vartheta(\boldsymbol{x})$ for all $\boldsymbol{x} \in \mathbb{R}^m$ implies $\theta = \vartheta$. Clearly, it is equivalent to address the uniqueness of the parametrization of V_θ, that is to show that relation $V_\theta(\boldsymbol{x}) = V_\vartheta(\boldsymbol{x})$ for all $\boldsymbol{x} \in \mathbb{R}^m$ implies $\theta = \vartheta$.

In this section, we address the question of identifiability for different instantiations of the general model (20):

- For the LR model,

$$V_\theta^{\text{LR}}(\boldsymbol{x}) = \gamma \sum_{i \in M} \omega_i\, u_i(x_i) + \beta \tag{22}$$

  where parameters θ belongs to $\Theta^{\text{LR}} = \{(\gamma, \beta, \boldsymbol{\omega}), \gamma > 0,\ \beta \in \mathbb{R}, \boldsymbol{\omega} \in \mathbb{R}^m \text{ with } \sum_{i \in M} \omega_i = 1\}$, and the local utility functions u_i are fixed. As the features are heterogeneous and given in different units, the aim of the local utility functions is to map the features onto a unique scale. A standard way to construct these utility functions in machine learning is to use a standardization or uniformization process given a dataset.

  For convenience, the range of the fixed local utility functions is supposed to include the unit interval $[0, 1]$. This is for instance the case with standardization or uniformization.
- For the CR model,

$$V_\theta^{\text{CR}}(\boldsymbol{x}) = \gamma\, C_v(u_1(x_i), \ldots, u_m(x_m)) + \beta \tag{23}$$

  where θ belongs to $\Theta^{\text{CR}} = \{(\gamma, \beta, v), \gamma > 0,\ \beta \in \mathbb{R},\ v : 2^M \to \mathbb{R}$ satisfying (4) and (5)$\}$, and the local utility functions u_i are fixed, as for LR. We also assume that the range of the fixed local utility functions includes the unit interval $[0, 1]$.
- For the UR model,

$$V_\theta^{\text{UR}}(\boldsymbol{x}) = \sum_{i \in M} u_i(x_i) + \beta \tag{24}$$

  Compared to the other models, parameter γ is amalgamated by the non-normalized local utilities $u_1, \ldots, u_m$. Following Section 4, we differentiate the cases where the local utility functions are monotone or non-monotone. In the first case, the parameters belong to $\Theta^{\text{UR}}_{\text{Mon}} = \{(\beta, u_1, \ldots, u_m),\ \beta \in \mathbb{R}$ and $u_1, \ldots, u_m$ fulfill (13), (14) and (15)$\}$. In the second case, the parameters belong to $\Theta^{\text{UR}}_{\text{NonMon}} = \{(\beta, u_1, \ldots, u_m),\ \beta \in \mathbb{R}$ and $u_1, \ldots, u_m$ satisfy (9), (10) and (11)$\}$.
- For the CUR model,

$$V_\theta^{\text{CUR}}(\boldsymbol{x}) = \gamma\, C_v(u_1(x_i), \ldots, u_m(x_m)) + \beta \tag{25}$$

  For monotone utility functions, the parameters belong to $\Theta^{\text{CUR}}_{\text{Mon}} = \{(\gamma, \beta, v, u_1, \ldots, u_m), \gamma > 0,\ \beta \in \mathbb{R},\ v : 2^M \to \mathbb{R}$ satisfying (4) and (5), and $u_1, \ldots, u_m$ fulfill (13), (14) and (16)$\}$. For non-monotone utility functions, the parameters belong to $\Theta^{\text{CUR}}_{\text{NonMon}} = \{(\gamma, \beta, v, u_1, \ldots, u_m), \gamma > 0,\ \beta \in \mathbb{R},\ v : 2^M \to \mathbb{R}$ satisfying (4) and (5), and $u_1, \ldots, u_m$ satisfy (9), (10) and (12)$\}$.

For a vector $\boldsymbol{x} = (x_1, \ldots, x_m)$ and a subset of indices $A \subset M$, we subsequently denote by $\boldsymbol{x}_A$ the projection of $\boldsymbol{x}$ to A and by $\boldsymbol{x}_{-A}$ the projection to the complement $M \setminus A$. Moreover, $(\boldsymbol{x}_A, \boldsymbol{y}_{-A})$ is the vector with entries $\boldsymbol{x}$ on A and $\boldsymbol{y}$ on $M \setminus A$. If A is a single index i, we simply write i and $-i$ instead of $\{i\}$ and $-\{i\}$.

5.1 Relationship between the four models

We show in this section that models LR, CR and UR are particular cases of CUR.

Proposition 1. *Models V_θ^{LR} with $\theta \in \Theta^{\mathrm{LR}}$, V_θ^{CR} with $\theta \in \Theta^{\mathrm{CR}}$, V_θ^{UR} with $\theta \in \Theta^{\mathrm{UR}}_{\mathrm{Mon}}$, V_θ^{UR} with $\theta \in \Theta^{\mathrm{UR}}_{\mathrm{NonMon}}$ and V_θ^{CUR} with $\theta \in \Theta^{\mathrm{CUR}}_{\mathrm{Mon}}$, can be represented by V_θ^{CUR} for some appropriate $\theta \in \Theta^{\mathrm{CUR}}_{\mathrm{NonMon}}$.*

Proof : Case of V_θ^{LR} with $\theta \in \Theta^{\mathrm{LR}}$. Let $\theta = (\gamma, \beta, \boldsymbol{\omega}) \in \Theta^{\mathrm{LR}}$. Then $V_\theta^{\mathrm{LR}} = V_{\theta'}^{\mathrm{CUR}}$ for $\theta' = (\gamma', \beta', v', u'_1, \ldots, u'_m)$, with $\gamma' = \gamma$, $\beta' = \beta$, $u'_1 = u_1$, …, $u'_m = u_m$ and $v'(S) = \sum_{i \in S} \omega_i$.

Case of V_θ^{CR} with $\theta \in \Theta^{\mathrm{CR}}$. Let $\theta = (\gamma, \beta, v) \in \Theta^{\mathrm{CR}}$. Then $V_\theta^{\mathrm{CR}} = V_{\theta'}^{\mathrm{CUR}}$ for $\theta' = (\gamma', \beta', v', u'_1, \ldots, u'_m)$, with $\gamma' = \gamma$, $\beta' = \beta$, $v' = v$ and $u'_1 = u_1$, …, $u'_m = u_m$.

Case of V_θ^{UR} with $\theta \in \Theta^{\mathrm{UR}}_{\mathrm{NonMon}}$.c Let $\theta = (\beta, u_1, \ldots, u_m) \in \Theta^{\mathrm{UR}}_{\mathrm{NonMon}}$. We set $u_i^\top = \sup_{x_i \in \mathbb{R}} u_i(x_i)$ for every $i \in M$. We have $0 < u_i^\top < \infty$ by (11).

Then $V_\theta^{\mathrm{UR}} = V_{\theta'}^{\mathrm{CUR}}$ for $\theta' = (\gamma', \beta', v', u'_1, \ldots, u'_m)$, with $\gamma' = \sum_{i \in M} u_i^\top$, $\beta' = \beta$, $u'_1 = \frac{u_1}{u_1^\top}$, …, $u'_m = \frac{u_m}{u_m^\top}$ and $v'(S) = \frac{\sum_{i \in S} u_i^\top}{\sum_{i \in M} u_i^\top}$.

Case of V_θ^{UR} with $\theta \in \Theta^{\mathrm{UR}}_{\mathrm{Mon}}$. We only have to use the previous case as $\Theta^{\mathrm{UR}}_{\mathrm{Mon}} \subset \Theta^{\mathrm{UR}}_{\mathrm{NonMon}}$.

Case of V_θ^{CUR} with $\theta \in \Theta^{\mathrm{CUR}}_{\mathrm{Mon}}$. The result is clear as $\Theta^{\mathrm{UR}}_{\mathrm{Mon}} \subset \Theta^{\mathrm{UR}}_{\mathrm{NonMon}}$. ■

5.2 Identifiability of the CUR model with non-monotone local utility functions

Let us start with the most general model, that is the CUR model with non-monotone local utility functions, according to Proposition 1. We consider thus model V_θ^{CUR}.

We assume that there is no useless criterion in V_θ^{CUR}:

$$\forall i \in M \; \exists x_i, y_i \in \mathbb{R}, \boldsymbol{z}_{-i} \in \mathbb{R}^{M \setminus \{i\}}, \quad V_\theta^{\mathrm{CUR}}(x_i, \boldsymbol{z}_{-i}) \neq V_\theta^{\mathrm{CUR}}(y_i, \boldsymbol{z}_{-i}) \qquad (26)$$

We introduce the following definition [10, 9].

Definition 1. *Consider a capacity v on M. A criterion $i \in M$ is said to be* degenerate *if $v(S \cup \{i\}) = v(S)$ for every $S \subseteq M \setminus \{i\}$. A capacity is said to be* non-degenerate *if there is no degenerate criterion.*

Then, our first result is as follows (adapted from [9]).

Lemma 1. *Relation (26) holds for V_θ^{CUR} for $\theta \in \Theta^{\mathrm{CUR}}_{\mathrm{NonMon}}$ if and only if $\gamma \neq 0$ and capacity v is non-degenerate.*

Proof : It is easy to see that if v is non-degenerate and $\gamma \neq 0$, then (26) holds as the range of the utility functions u_i is $[0, 1]$. Moreover, if $\gamma = 0$ then (26) cannot be true.

Assume now that $\gamma \neq 0$ and v is degenerate with respect to a given attribute i. Consider any $z_{-i} \in \mathbb{R}^{M \setminus \{i\}}$, and $x_i, y_i \in \mathbb{R}$ with $x_i \neq y_i$. Let $\tau : \{1, \ldots, n-1\} \to M \setminus \{i\}$ such that

$$u_{\tau(1)}(z_{\tau(1)}) \leq u_{\tau(2)}(z_{\tau(2)}) \leq \cdots \leq u_{\tau(n-1)}(z_{\tau(n-1)}).$$

Moreover, let k_x and k_y such that $u_{\tau(k_x)}(z_{\tau(k_x)}) \leq u_i(x_i) \leq u_{\tau(k_x+1)}(z_{\tau(k_x+1)})$ and $u_{\tau(k_y)}(z_{\tau(k_y)}) \leq u_i(y_i) \leq u_{\tau(k_y+1)}(z_{\tau(k_y+1)})$. Without loss of generality, we assume that $k_y \leq k_x$. Then

$$\begin{aligned}
&V(x_i, z_{-i}) - V(y_i, z_{-i}) \\
&= \sum_{l=k_y+1}^{k_x} u_{\tau(l)}(z_{\tau(l)}) \\
&\Big[\Big(\mu(\{i, \tau(l), \ldots, \tau(n-1)\}) - \mu(\{i, \tau(l+1), \ldots, \tau(n-1)\})\Big) \\
&- \Big(\mu(\{\tau(l), \ldots, \tau(n-1)\}) - \mu(\{\tau(l+1), \ldots, \tau(n-1)\})\Big)\Big] \\
&+ u_i(x_i) \Big(\mu(\{i, \tau(k_x+1), \ldots, \tau(n-1)\}) \\
&\qquad - \mu(\{\tau(k_x+1), \ldots, \tau(n-1)\})\Big) \\
&+ u_i(y_i) \Big(\mu(\{i, \tau(k_y+1), \ldots, \tau(n-1)\}) \\
&\qquad - \mu(\{\tau(k_y+1), \ldots, \tau(n-1)\})\Big)
\end{aligned}$$

This sum is equal to zero as v is degenerate with respect to i. Hence, (26) cannot be fulfilled. ■

The following result shows the uniqueness of the representation of the CUR model V_θ^{CUR} for θ in $\Theta^{\mathrm{CUR}}_{\mathrm{NonMon}}$.

Proposition 2. *Consider a continuous utility function $V : \mathbb{R}^M \to \mathbb{R}$ satisfying (26). For $\theta, \vartheta \in \Theta^{\mathrm{CUR}}_{\mathrm{NonMon}}$, relation $V_\theta^{\mathrm{CUR}}(\boldsymbol{x}) = V_\vartheta^{\mathrm{CUR}}(\boldsymbol{x}) = V(\boldsymbol{x})$ for all $\boldsymbol{x} \in \mathbb{R}^m$ implies $\theta = \vartheta$. Moreover the related capacity v in θ, ϑ is non-degenerate.*

Proof : We prove the claim by contradiction. Thus, assume $V(\boldsymbol{x}) = \gamma\, C_v(u_1(x_1), \ldots, u_m(x_m)) + \beta$ and $V'(\boldsymbol{x}) = \gamma'\, C_{v'}(u'_1(x_1), \ldots, u'_m(x_m)) + \beta'$ are equal for all $\boldsymbol{x} \in \mathbb{R}^M$, with $\theta = (\gamma, \beta, v, u_1, \ldots, u_m) \neq (\gamma', \beta', v', u'_1, \ldots, u'_m) = \vartheta$. By Lemma 1, capacities v and v' are non-degenerate.

Let us define reference values $\underline{x}_i \in \underline{A}_i$, $\underline{x}'_i \in \underline{A}'_i$, $\overline{x}_i \in \overline{A}_i$ and $\overline{x}'_i \in \overline{A}'_i$.

Part 1: Assume first that $u_1 = u'_1, \ldots, u_m = u'_m$. Then we can choose the reference values so that $\underline{x}_i = \underline{x}'_i$ and $\overline{x}_i = \overline{x}'_i$ for all $i \in M$. For every $S \subseteq M$, we have

$$\begin{aligned}
&\lim_{\boldsymbol{x}_S \to \overline{\boldsymbol{x}}_S} \lim_{\boldsymbol{x}_{N \setminus S} \to \underline{\boldsymbol{x}}_{N \setminus S}} V(\boldsymbol{x}_S, \boldsymbol{x}_{N \setminus S}) = \gamma\, C_v(1_S, 0_{N \setminus S}) + \beta \\
&= \gamma\, v(S) + \beta \\
&= \lim_{\boldsymbol{x}_S \to \overline{\boldsymbol{x}}_S} \lim_{\boldsymbol{x}_{N \setminus S} \to \underline{\boldsymbol{x}}_{N \setminus S}} V'(\boldsymbol{x}_S, \boldsymbol{x}_{N \setminus S}) = \gamma'\, v'(S) + \beta'
\end{aligned}$$

According to (4), the previous relation for $S = \emptyset$ and $S = M$ gives $\beta = \beta'$ and $\gamma = \gamma'$. We obtain thus $v = v'$, which contradicts our assumption. Hence, there is at least an index $k \in M$ such that $u_k \neq u'_k$.

Part 2: Assume now that $u_k \neq u'_k$ and $[\underline{A}_k \cap \underline{A}'_k = \emptyset$ or $\overline{A}_k \cap \overline{A}'_k = \emptyset]$ for some $k \in M$. Then, there exists an interval $[x_k, y_k]$ not reduced to a single value such that u_k and u'_k do not have the same monotonicity. Without loss of generality, assume that u_k is strictly increasing in $[x_k, y_k]$ and u'_k is non-increasing in $[x_k, y_k]$: $u_k(x_k) < u_k(y_k)$ and $u'_k(x_k) \geq u'_k(y_k)$.

As capacity v is non-denegerate, there exists $S \subseteq N$ with

$v(S \cup k) > v(S)$. Then, one can readily see that

$$\begin{aligned}
&\lim_{\boldsymbol{x}_S \to \overline{\boldsymbol{x}}_S} \lim_{\boldsymbol{x}_{N\setminus(S\cup\{k\})} \to \underline{\boldsymbol{x}}_{N\setminus(S\cup\{k\})}} V(x_k, \boldsymbol{x}_S, \boldsymbol{x}_{N\setminus(S\cup\{k\})}) \\
&= \gamma\, C_v(u_k(x_k), 1_S, 0_{N\setminus(S\cup\{k\})}) + \beta \\
&= \gamma \left[u_k(x_k)\,(v(S\cup k) - v(S)) + v(S)\right] + \beta \\
&< \gamma \left[u_k(y_k)\,(v(S\cup k) - v(S)) + v(S)\right] + \beta \\
&= \gamma\, C_v(u_k(y_k), 1_S, 0_{N\setminus(S\cup\{k\})}) + \beta \\
&= \lim_{\boldsymbol{x}_S \to \overline{\boldsymbol{x}}_S} \lim_{\boldsymbol{x}_{N\setminus(S\cup\{k\})} \to \underline{\boldsymbol{x}}_{N\setminus(S\cup\{k\})}} V(y_k, \boldsymbol{x}_S, \boldsymbol{x}_{N\setminus(S\cup\{k\})})
\end{aligned}$$

On the other hand, by monotonicity conditions (5) and $u'_k(x_k) \geq u'_k(y_k)$,

$$\begin{aligned}
&\lim_{\boldsymbol{x}_S \to \overline{\boldsymbol{x}}_S} \lim_{\boldsymbol{x}_{N\setminus(S\cup\{k\})} \to \underline{\boldsymbol{x}}_{N\setminus(S\cup\{k\})}} V'(x_k, \boldsymbol{x}_S, \boldsymbol{x}_{N\setminus(S\cup\{k\})}) \\
&\geq \lim_{\boldsymbol{x}_S \to \overline{\boldsymbol{x}}_S} \lim_{\boldsymbol{x}_{N\setminus(S\cup\{k\})} \to \underline{\boldsymbol{x}}_{N\setminus(S\cup\{k\})}} V'(y_k, \boldsymbol{x}_S, \boldsymbol{x}_{N\setminus(S\cup\{k\})})
\end{aligned}$$

We obtain a contradiction thanks to the continuity of V and to relations $V(x_k, \boldsymbol{x}_S, \boldsymbol{x}_{N\setminus(S\cup\{k\})}) = V'(x_k, \boldsymbol{x}_S, \boldsymbol{x}_{N\setminus(S\cup\{k\})})$ and $V(y_k, \boldsymbol{x}_S, \boldsymbol{x}_{N\setminus(S\cup\{k\})}) = V'(y_k, \boldsymbol{x}_S, \boldsymbol{x}_{N\setminus(S\cup\{k\})})$.

Part 3: There remains the case where $\underline{A}_i \cap \underline{A}'_i \neq \emptyset$ and $\overline{A}_i \cap \overline{A}'_i \neq \emptyset$ for all $i \in M$, but $u_k \neq u'_k$ for some $k \in M$. Then, we can choose the reference values so that $\underline{a}_i = \underline{a}'_i$ and $\overline{a}_i = \overline{a}'_i$ for all $i \in M$. As capacity v is non-degenerate, there exists $S \subseteq M$ with $v(S \cup k) > v(S)$. For every $x_k \in \mathbb{R}$, we write

$$\begin{aligned}
&\lim_{\boldsymbol{x}_S \to \overline{\boldsymbol{x}}_S} \lim_{\boldsymbol{x}_{N\setminus(S\cup\{k\})} \to \underline{\boldsymbol{x}}_{N\setminus(S\cup\{k\})}} V(x_k, \boldsymbol{x}_S, \boldsymbol{x}_{N\setminus(S\cup\{k\})}) \\
&= \gamma\, C_v(u_k(x_k), 1_S, 0_{N\setminus(S\cup\{k\})}) + \beta \\
&= \gamma \left[u_k(x_k)\,(v(S\cup k) - v(S)) + v(S)\right] + \beta \\
&= \lim_{\boldsymbol{x}_S \to \overline{\boldsymbol{x}}_S} \lim_{\boldsymbol{x}_{N\setminus(S\cup\{k\})} \to \underline{\boldsymbol{x}}_{N\setminus(S\cup\{k\})}} V'(x_k, \boldsymbol{x}_S, \boldsymbol{x}_{N\setminus(S\cup\{k\})}) \\
&= \gamma'\, C_{v'}(u'_k(x_k), 1_S, 0_{N\setminus(S\cup\{k\})}) + \beta' \\
&= \gamma' \left[u'_k(x_k)\,(v'(S\cup k) - v'(S)) + v'(S)\right] + \beta'
\end{aligned}$$

Hence, $\gamma\,(v(S\cup k) - v(S))\,u_k(x_k) + \left[\gamma\, v(S) + \beta\right] = \gamma'\,(v'(S\cup k) - v'(S))\,u'_k(x_k) + \left[\gamma'\, v'(S) + \beta'\right]$ for every $x_k \in \mathbb{R}$. Thus, u_k is equal to u'_k up to an affine transformation: $u'_k(x_k) = r\,u_k(x_k) + t$. As $\lim_{x_k \to \underline{x}_k} u_k(x_k) = 0 = \lim_{x_k \to \underline{x}_k} u'_k(x_k)$ and $\lim_{x_k \to \overline{x}_k} u_k(x_k) = 1 = \lim_{x_k \to \overline{x}_k} u'_k(x_k)$, we conclude that $u_k = u'_k$. Hence, we again obtain a contradiction. ■

5.3 Identifiability for the other models

The following result derives from the previous main proposition, the uniqueness of the representation of the other models.

Corollary 1. *We have the following results:*

- *The representation of the LR model V_θ^{LR}, with $\theta \in \Theta^{\mathrm{LR}}$, is unique.*
- *The representation of the CR model V_θ^{CR}, with $\theta \in \Theta^{\mathrm{CR}}$, is unique.*
- *The representation of a continuous UR model V_θ^{UR}, with $\theta \in \Theta^{\mathrm{UR}}_{\mathrm{Mon}}$, is unique.*
- *The representation of a continuous UR model V_θ^{UR}, with $\theta \in \Theta^{\mathrm{UR}}_{\mathrm{NonMon}}$, is unique.*
- *Consider a continuous utility function $V : \mathbb{R}^M \to \mathbb{R}$ satisfying (26). For $\theta, \vartheta \in \Theta^{\mathrm{CUR}}_{\mathrm{Mon}}$, relation $V_\theta^{\mathrm{CUR}}(\boldsymbol{x}) = V_\vartheta^{\mathrm{CUR}}(\boldsymbol{x}) = V(\boldsymbol{x})$ for all $\boldsymbol{x} \in \mathbb{R}^m$ implies $\theta = \vartheta$. Moreover the related capacity v in θ, ϑ is non-degenerate.*

Proof : The corollary basically follows from Propositions 2 and 1.

Assumption (26) is necessary to ensure uniqueness only for model CUR. Indeed, for this model, if a feature i is degenerate, then changing the corresponding local utility function u_i will not modify V_θ. For the other models, uniqueness is ensured even if the a feature is degenerate. This is in particular true for the UR model as we do not separate local utilities from weights. Besides, degeneracy of features is ruled out by conditions (10) and (11).

Continuity of the global model V_θ is necessary for UR and CUR, but not for LR and CR. Indeed for these two latter models, we fall under Part 1 of the proof of Proposition 2 ($u_1 = u'_1, \ldots, u_m = u'_m$). Continuity of V is necessary as the reference elements $\underline{x_i}$ and $\overline{x}_i$ may be infinite. For models LR and CR, the reference elements $\underline{x_i}$ and $\overline{x}_i$ do not necessarily have to belong to $\underline{A}_i$ and $\overline{A}_i$. We have only to ensure that $u_1(\underline{x}_1) = \cdots = u_m(\underline{x}_m)$ and $u_1(\overline{x}_1) = \cdots = u_m(\overline{x}_m)$. These onditions can be fulfilled with finite values of $\underline{x_i}$ and $\overline{x}_i$. In this case, limits and thus continuity of V are no more necessary. ■

6 Conclusion and Future Work

The idea of preference learning based on "choquistic" models, combining the Choquet integral for representing (global) utility with logistic regression as a basic machine learning method, was put forward in [16, 5]. In these works, however, the local utility functions were assumed to be given beforehand and not subject to data-driven adaptation. A first step toward generalizing this approach was made in [19], where both the Choquet integral (capacity) and the local utilities were learned simultaneously, albeit the latter being restricted to monotone functions.

In any case, if local and global utility functions are both paramaterized and learned at the same time, it is no longer clear that the overall model remains identifiable. In this paper, we have therefore addressed the question of identifiability and given affirmative answers for important classes of models, including both monotne and non-monotone local utility functions.

On the one side, our results provide a solid basis for developing more sophisticated practical learning methods. In particular, we plan to extend the approach presented in [19] from monotone to non-monotone local utility functions.

On the other side, there is also scope for further extensions of the theoretical analysis. For example, relaxing the restriction to the case of two decisions ($y \in \{0, 1\}$), we plan to extend our results to the more general case of ranking on an ordinal scale, which is directly connected to ordinal logistic (choquistic) regression [18].

REFERENCES

[1] G. Choquet. Theory of capacities. *Annales de l'Institut Fourier*, 5:131–295, 1953.

[2] J. Fürnkranz and E. Hüllermeier, editors. *Preference Learning*. Springer-Verlag, 2010.

[3] M. Grabisch. The application of fuzzy integrals in multicriteria decision making. *European J. of Operational Research*, 89:445–456, 1996.

[4] M. Grabisch and Ch. Labreuche. A decade of application of the Choquet and Sugeno integrals in multi-criteria decision aid. *Annals of Operation Research*, 175:247–286, 2010.

[5] E. Hüllermeier and A. Fallah Tehrani. Efficient learning of classifiers based on the 2-additive Choquet integral. In C. Moewes and A. Nürnberger, editors, *Computational Intelligence in Intelligent Data Analysis*, Studies in Computational Intelligence, pages 17–30. Springer, 2012.

[6] E. Jacquet-Lagrèze and Y. Siskos. Assessing a set of additive utility functions for multicriteria decision making: The UTA method. *European J. of Operational Research*, 10:151–164, 1982.

[7] R. L. Keeney and H. Raiffa. *Decision with Multiple Objectives*. Wiley, New York, 1976.

[8] D.H. Krantz, R.D. Luce, P. Suppes, and A. Tversky. *Foundations of measurement*, volume 1: Additive and Polynomial Representations. Academic Press, 1971.

[9] Ch. Labreuche. An axiomatization of the choquet integral and its utility functions without any commensurability assumption. *submitted*.

[10] Ch. Labreuche. An axiomatization of the Choquet integral and its utility functions without any commensurateness assumption. In *Int. Conf. on Information Processing and Management of Uncertainty in Knowledge-Based Systems (IPMU)*, Catania, Italy, July 9-13 2012.

[11] Patrice Perny, Paolo Viappiani, and Abdellah Boukhatem. Incremental preference elicitation for decision making under risk with the rank-dependent utility model. In *Uncertainty in Artificial Intelligence*, New York, United States, 2016.

[12] M. Pirlot, H. Schmitz, and P. Meyer. An empirical comparison of the expressiveness of the additive value function and the choquet integral models for representing rankings. In *25th Mini-EURO Conference Uncertainty and Robustness in Planning and Decision Making (URPDM 2010)*, pages 374–387, Coimbra, Portugal, April 2010.

[13] Olivier Sobrie, Vincent Mousseau, and Marc Pirlot. Learning a majority rule model from large sets of assignment examples. In *Proc. ADT, 3rd International Conference on Algorithmic Decision Theory*, pages 336–350, Bruxelles, Belgium, 2013.

[14] Olivier Sobrie, Vincent Mousseau, and Marc Pirlot. Learning the parameters of a non compensatory sorting model. In *Proc. ADT, 4th International Conference on Algorithmic Decision Theory*, pages 153–170, Lexington, KY, USA, 2015.

[15] M. Sugeno. *Theory of fuzzy integrals and its applications*. PhD thesis, Tokyo Institute of Technology, 1974.

[16] A. Fallah Tehrani, W. Cheng, K. Dembczynski, and E. Hüllermeier. Learning monotone nonlinear models using the choquet integral. *Machine Learning*, 89:183–211, 2012.

[17] A. Fallah Tehrani, W. Cheng, and E. Hüllermeier. Preference learning using the Choquet integral: The case of multipartite ranking. *IEEE Transactions on Fuzzy Systems*, 20(6):1102–1113, 2012.

[18] A. Fallah Tehrani and E. Hüllermeier. Ordinal Choquistic regression. In J. Montero, G. Pasi, and D. Ciucci, editors, *Proceedings EUSFLAT–2013, 8th International Conference of the European Society for Fuzzy Logic and Technology*, Milano, Italy, 2013. Atlantis Press.

[19] A. Fallah Tehrani, Ch. Labreuche, and E. Hüllermeier. Choquistic utilitaristic regression. In *Decision Aid to Preference Learning (DA2PL) workshop*, Chatenay-Malabry, France, November 2014.

Accountable classification without frontiers

K. Belahcene[1], C. Labreuche[2], N. Maudet[3], V. Mousseau[1], W. Ouerdane[1]

Abstract. This paper addresses the problem of multicriteria ordinal sorting through the lens of *accountability*, the ability of a human decision maker to own a recommendation made by the system. It proposes a number of model features that would favor the capability to support the recommendation with a convincing explanation, builds the only model possessing them all, discusses its aptitude at correctly representing some given preference information, and thoroughly describes the output recommendation, from mathematical and computational points of view. This recommendation is supported by an *explanation*, that takes the form of arguments implementing some specific *argument schemes*. Finally, it uncovers some flaws of the model, analyses their consequences, and proposes several directions for future research.

1 Introduction

Multi Criteria Decision Analysis (MCDA) aims at helping a decision maker in deciding between options described according various points of view, formally represented by numerical functions called *criteria*. In this paper, this decision is modeled as an *ordinal sorting problem*, where options are to be assigned to some class among an ordered set of classes. Many algorithms have been proposed since 50 years to address this classification problem, usually focusing on *accuracy* or *velocity*, performance metrics hardly suited to the context of MCDA, where user satisfaction is hard to measure, and is sought for by using "interpretable, cognitively plausible decision models" that would yield "convincing explanations of decisions". This vague statement of principle does not seem to be sufficient by itself, as even models often thought to be self-explanatory, such as decision trees, provide a trace of their execution which can be shown to not be a good support for explanation [1]. Hence, we try to take this principle one step further by introducing the notion of *accountability* - the ability of a human decision maker to own a recommendation made by the system and actually implement it - and addressing it by providing *explanations*.

This requirement for *accountability* is more demanding than the mere combination of simpler goals such as *trust*, *transparency*, or *persuasiveness* [43], as its litmus test is the ability of the recipient of the recommendation to defend it in front of other, skeptical, stakeholders of the decision. *Trust* requires the recommendation to be consistently accurate, but eventually asks for delegation of the decision to the system. *Transparency* translates to a formal knowledge of the underlying algorithm, not an actionable one. *Persuasiveness* is hardly transferable: someone persuaded by the recommendation (or its presentation) is not necessarily a good persuader.

In the specific case of an ordinal sorting problem, we outline various features of a model that would presumably enhance its accountability (Section 2). By formalizing these requirements, we design the only possible model satisfying them all, decomposing it in a learning phase (Section 3) and a recommendation phase (Section 4). We provide formal explanations of the recommendation in most cases, in the form of pieces of preference information arranged according to specific *argument schemes* tailored to represent the specific rules of the model. Section 5 concludes the paper, by putting its findings into perspective.

2 Design principles

We aim at building an accountable, ordinal, multicriteria classifier, mapping a *candidate* object to a *recommendation* consisting in one or more category among a predefined, ordered collection of these. In a multicriteria context, the only indisputable relation between objects is the Pareto dominance, occurring when an object outperforms another on every criteria. As the situation is seldom so clear, the rules permitting the comparison of objects need to be enriched, taking into account the knowledge, values, opinions, beliefs, needs, ... of the decision maker, collected under the label *preference information*, which is also considered an input of the classifier. We also consider an additional output, an *explanation* aimed at the decision maker, supporting the recommendation and hopefully enabling the accountability sought for.

In order to reach this goal of accountability, we make several strong, additional assumptions about the recommender system, described in the following sections.

2.1 No jargon

As the need for accountability proceeds from issues in the human - machine interface at the outer end of the model, we ought to take them into account at its inner end. The natural terms in which the decision maker usually expresses her preferences are by making assignments of prototypical alternatives to categories. Consequently, in order to accurately represent the specific decision process , we opt for an indirect elicitation mechanism [17, 39, 13]: the decision maker is never asked any questions about artifacts of the model (weights, profiles, coefficients, thresholds,...); instead she should express preferences directly in the language of the actual decision situation.

2.2 No values

Quite naturally, the requirement of accountability invited us to narrow the focus to the models used in MCDA, and particularly those based on outranking methods, as they are specifically intended to

[1] LGI, CentraleSupélec, Université Paris-Saclay, Chatenay Malabry, France; emails: {khaled.belahcene, vincent.mousseau, wassila.ouerdane}@centralesupelec.fr
[2] Thales Research & Technology, 91767 Palaiseau Cedex, France; email: christophe.labreuche@thalesgroup.com
[3] Sorbonne Universités, UPMC Univ Paris 06, CNRS, LIP6 UMR 7606, 75005 Paris; email: nicolas.maudet@lip6.fr

account for the observed peculiarities of the human reasoning with respect to trade-offs. These methods focus on the representation of an *outranking relation* [45, 8, 36], a binary relation between the objects being classified, formulating the (weak) preference of an object over another. This set of pairwise comparisons offers a wider palette than the classical aggregation to a single value, such as described and implemented by the MAVT [23] Also, *outranking relations* are able to model non-transitive preferences, which is not possible with scoring methods.

2.3 No frontiers

Some classifiers explicitly refer to frontiers between categories, especially in the field of MCDA. Models such as ELECTRE TRI [48, 37], MR-Sort [29], NCS [11], UTADIS [16], the categories are separated by limiting profiles. For value-based models, limiting profiles are scalars, while outranking models use vector profiles. Elicitation, i.e. fitting these parameters to the decision situation considered, is a demanding affair. It often requires to collect a lot of preference information from the decision maker, and a considerable computational effort. In our quest for accountability, we investigate the opportunity to do without this construct. Several works have already explored this venue, such as the models built using the logical analysis of data (LAD) techniques [15, 9], or ORCLASS [28], which output classification rules, or the outranking models proposed by [25] or [24]. We propose the following rules, directly linking classification to the relative position with respect to reference objects:

- **Rule 1:** an object can not outrank any object assigned to a strictly better class;
- **Rule 2:** an object outranks objects assigned to a strictly worse class;
- **Rule 3:** objects in the same class can be in any position with respect to outranking.

2.4 No compensation

We opt for a preference model where pairwise comparisons are adjudicated by considering solely the *set* of criteria for which one alternative is at least as good as the other, not the specific values of these criteria. Consequently, being very good on some criterion can not compensate for low performance on others. This feature enables the algorithm to proceed without performing any algebraic computation, which makes it particularly suited to a dialectical context. It is shared with established *non-compensatory* [18, 12] ordinal sorting models used in the field of MCDA: ELECTRE TRI, MR-Sort, NCS.

2.5 No inference

An approach used by many models, sometimes called *inference*, consists in selecting a specific array of parameters, usually by maximizing some secondary objective over the set of parameters consistent with the preference information [38, 10, 19]. This is not a venue we choose to explore. Instead, we opt for an approach sometimes qualified as *robust* (to the lack of preference information) [46, 22], formulating a - possibly partial - recommendation that can not be refuted by any judgment function consistent with the preference information.

3 Formal description

In this section, we translate the problem description and design principles of the previous section into a mathematical framework, with a formal description of the relevant conceits as well as theorems describing their properties.

3.1 The recommender system

We consider a multicriteria ordinal sorting problem : a collection of objects are evaluated on a set of criteria N. We note $\mathbb{B} := \{0, 1\}$, so that elements of $\mathbb{B}^N$ are at the same time vectors with binary coordinates, and subsets of N, partially ordered by inclusion. The maximal element of $\mathbb{B}^N$ is the unanimous coalition N, also denoted $(1, \dots, 1)$. The minimal element of $\mathbb{B}^N$ is the empty coalition $\emptyset$, also denoted $(0, \dots, 0)$. Each criterion $i \in N$ maps an object to a performance value in a totally ordered set $\mathbb{X}_i$, the higher the better. Consequently, each object is described by a performance vector in the partially ordered set $\mathbb{X} = \prod_{i \in N} \mathbb{X}_i$. The objects are to be assigned to some class chosen among an ordered set $\widehat{C} = \{C_1 \prec \cdots \prec C_p\}$, so that assignment to a class with a high index is desirable.

Formally, let us describe the recommender system as a function mapping a pair $< z, \mathcal{P} >$ to a pair $< K, \mathcal{E} >$, where:

- The object $z \in \mathbb{X}$ is a *candidate* for sorting;
- $\mathcal{P}$ denotes *preference information* collected from the decision maker consisting of typical classification examples, a collection of *reference objects* $\mathbb{X}^\star \subset \mathbb{X}$, and their assigned categories $Class : \mathbb{X}^\star \to \widehat{C}$. For syntactic reasons, we represent it by a set of object-assignment pairs $\mathcal{P} \subset \mathbb{X} \times \widehat{C}$.

$$\mathcal{P} := \bigcup_{x^\star \in \mathbb{X}^\star} (x^\star, Class(x^\star))$$

- $K \subset \widehat{C}$ is the *recommendation*, concerning the classes that could be assigned to the candidate, detailed in Section 4;
- $\mathcal{E}$ is an *explanation* yet unspecified, supporting the recommendation K (see for instance [27, 7]), and addressed by Section 4.

Example 1. *Objects are evaluated according to four criteria:*

- *a, valued on the ordinal scale* $\mathbb{X}_a := \{A > B > C\}$*;*
- *b, valued on the ordinal scale* $\mathbb{X}_b := \{A > B > C\}$*;*
- *c, valued on the cardinal scale* $\mathbb{X}_c := [0, 5]$*, higher is better;*
- *d, valued on the binary scale* $\mathbb{X}_d := \{True > False\}$*;*

Six reference objects: $\mathbb{X}^\star := \{A_1, A_2, B_1, B_2, C_1, C_2\}$*, described by the performance table below, are assigned to three classes:* $\widehat{C} := \{\star \prec \star\star \prec \star\star\star\}$ *and make up the preference information* $\mathcal{P}$*. We consider three candidates :* Z_1, Z_2, Z_3 *and try to assign them to some possible classes.*

$Object$	a	b	c	d	$Assignment$
A_1	A	A	2.5	$False$	$\star\star\star$
A_2	A	B	2.1	$True$	$\star\star\star$
B_1	B	B	1.3	$True$	$\star\star$
B_2	A	C	3.7	$False$	$\star\star$
C_1	B	C	1.6	$True$	$\star$
C_2	C	C	4.1	$False$	$\star$
Z_1	B	B	1.1	$False$	?
Z_2	B	A	1.8	$False$	??
Z_3	A	B	1.2	$False$	???

3.2 The reasoning of the decision maker

A non-compensatory outranking relation can be represented by a Boolean composite function:

$$\forall x, y \in \mathbb{X},\ xS_\phi y \iff \phi \circ O_N(x,y) = 1$$

where the *observation function* O_N maps a pair of objects to its *concordance set*, and the consistent judgment of the decision maker, based on these concordance sets, is represented by the *judgment function* ϕ mapping a concordance set to a truth value.

$$\begin{array}{rccl} O_N: & \mathbb{X}\times\mathbb{X} & \to & \mathbb{B}^N \\ & (x,y) & \mapsto & \{i \in N : x_i \geq y_i\} \end{array}$$

Antecedents of 1 by ϕ, called *true points* in the language of the LAD [15], represent *sufficient coalitions of criteria*, while antecedents of 0 by ϕ are *false points* or *insufficient coalitions of criteria*. ϕ is supposed *non-decreasing*, meaning that a superset of a sufficient coalition of criteria is also sufficient, and a subset of an insufficient coalition is also insufficient. Compatibility of the outranking relation S to the Pareto dominance imposes that a unanimous support of criteria is always sufficient, so $\phi(N) = 1$. Conversely, $\phi(\emptyset) = 0$ must hold, so the relation S is not reduced to generalized indifference.

Finally, we define the set of any possible judgment function :

$$\phi \in \widehat{\Phi} := \{\phi : \mathbb{B}^N \to \mathbb{B} \ : \phi \nearrow \text{ and } \phi(N) = 1 \text{ and } \phi(\emptyset) = 0\}$$

3.3 Learning from the assignment examples

We denote $\succsim_\mathcal{P}$ the complete preorder between reference objects induced by $\mathcal{P}$:

$$\left\{ \begin{array}{lcl} x^\star \succsim_\mathcal{P} y^\star & \iff & Class(x^\star) \succsim Class(y^\star) \\ x^\star \succ_\mathcal{P} y^\star & \iff & Class(x^\star) \succ Class(y^\star) \end{array} \right.$$

We consider the strict enforcement of the model rules for reference objects:

- Rule 1 : $\forall x^\star, y^\star \in \mathbb{X}^\star, x^\star \succ_\mathcal{P} y^\star \Rightarrow Not\ (y^\star S_\phi x^\star)$;
- Rule 2 : $\forall x^\star, y^\star \in \mathbb{X}^\star, x^\star \succ_\mathcal{P} y^\star \Rightarrow x^\star S_\phi y^\star$.

Hence, the assignment of reference objects expressed by $\mathcal{P}$ places upper (by Rule 1) and lower (by Rule 2) bounds upon the outranking relation between reference objects. so that:

$$\succ_\mathcal{P} \subset S_\phi \cap (\mathbb{X}^\star)^2 \subset \succsim_\mathcal{P}$$

These constraints transfer to the judgment functions. Each pair $(x^\star, y^\star)$ is mapped by the observation function O_N to a coalition of criteria. The observed coalitions $O_N(\mathbb{X}^\star \times \mathbb{X}^\star)$ serve as a learning set for the judgment function ϕ. They are sorted between three sets, yielding necessary conditions on ϕ:

- insufficient coalitions $O_N(\prec_\mathcal{P})$ should be mapped to 0 by ϕ;
- sufficient coalitions $O_N(\succ_\mathcal{P})$ should be mapped to 1 by ϕ;
- $O_N(\sim_\mathcal{P})$, which images by ϕ are not constrained.

Consequently, we define the set $\Phi(\mathcal{P})$ of judgment functions compatible to the preference information $\mathcal{P}$:

$$\Phi(\mathcal{P}) := \{\phi \in \widehat{\Phi} \ :\ \phi \circ O_N(\succ_\mathcal{P}) = 1 \text{ and } \phi \circ O_N(\prec_\mathcal{P}) = 0\}$$

Example 2. *(ex. 1 continued) In the following table, we detail all the relevant observed coalitions. Sufficient coalitions appear in the upper right side, boldfaced, while insufficient coalitions are in the lower left side.* ***N*** *stands for unanimity, which is self-explanatory.*

	$\star\star\star$		$\star\star$		$\star$	
	A_1	A_2	B_1	B_2	C_1	C_2
A_1	–	–	**abc**	**abd**	**abc**	**abd**
A_2	–	–	**N**	**abd**	**abc**	**abd**
B_1	d	bd	–	–	**abd**	**abd**
B_2	acd	ac	–	–	**abc**	**abd**
C_1	d	d	acd	bd	–	–
C_2	cd	c	c	bcd	–	–

3.4 Consistency of judgment

The set $\Phi(\mathcal{P})$ is empty if, and only if, Pareto dominance is contradicted ($\exists\ x^\star, y^\star \in \mathbb{X}^\star,\ \forall i \in N,\ x_i^\star \geq y_i^\star$ and $Class(x^\star) < Class(y^\star)$), or some coalition of criteria $M \in \mathbb{B}^N$ observed as being sufficient is weaker (for inclusion) than some coalition $M' \in \mathbb{B}^N$ observed as being insufficient. In such a case, we call the preference information $\mathcal{P}$ *inconsistent*; otherwise, it is consistent and $\Phi(\mathcal{P})$ is a *partially defined Boolean function* [15] . Combining the constraints on the judgment functions expressed by $\widehat{\Phi}$ and by $\mathcal{P}$, we can compute the true points of $\Phi(\mathcal{P})$. They are the antecedents of 1 common to every judgment function $\phi \in \Phi(\mathcal{P})$, and represent the coalitions *established as sufficient*, by the virtue of being at least as strong as an observed sufficient coalition.

$$\mathcal{T}_\mathcal{P} := \{t \in \mathbb{B}^N : \exists\, t_{obs} \in O_N(\succ_\mathcal{P}),\ t_{obs} \subset t\}$$

Conversely, the false points are the antecedents of zero common to every $\phi \in \Phi(\mathcal{P})$ and represent the coalitions established as insufficient.

$$\mathcal{F}_\mathcal{P} := \{f \in \mathbb{B}^N : \exists\, f_{obs} \in O_N(\prec_\mathcal{P}),\ f_{obs} \supset f\}$$

Proposition 1 details three manners to express inconsistency:

Proposition 1. *For any $\mathcal{P} \subset \mathbb{X} \times \widehat{C}$, the three following conditions are equivalent and characterize inconsistency :*

1. *Absence of compatible judgment function: $\Phi(\mathcal{P}) = \emptyset$*
2. *Conflicting constraints: $\mathcal{T}_\mathcal{P} \cap \mathcal{F}_\mathcal{P} \neq \emptyset$*
3. *Explicit contradiction: $\exists t \in O_N(\succ_\mathcal{P}), \exists f \in O_N(\prec_\mathcal{P}) : t \subset f$*

Example 3. *(ex. 2 continued) Coalitions are sorted according to the observations, and monotonicity:*

$O_N(\succ_\mathcal{P}) = \{N, abc, abd\} = \mathcal{T}_\mathcal{P}$

$O_N(\prec_\mathcal{P}) = \{c, d, ac, bd, cd, acd, bcd\}$

$\mathcal{F}_\mathcal{P} = \{\emptyset, a, b, c, d, ac, ad, bc, bd, acd, bcd\}$

There is no dispute, as $\mathcal{T}_\mathcal{P} \cap \mathcal{F}_\mathcal{P} = \emptyset$, but the coalition ab is left undecided.

4 Recommendations and explanations

In the previous section, we saw how the decision maker interprets pairwise comparisons between reference objects belonging to different classes as sufficient or insufficient coalitions of criteria. Here comes a new candidate, $z \in \mathbb{X}$. It gauges every reference object in $\mathbb{X}^\star$, yielding $|\mathcal{P}|$ observations $\overrightarrow{o}(z,\mathcal{P}) := \bigcup_{x^\star \in \mathbb{X}^\star} O_N(z, x^\star)$, and is also evaluated by every reference object, yielding $|\mathcal{P}|$ other observations $\overleftarrow{o}(z,\mathcal{P}) := \bigcup_{x^\star \in \mathbb{X}^\star} O_N(x^\star, z)$. Each of these $2|\mathcal{P}|$ observations is interpreted as a *sufficient, insufficient* or *undecided* coalition of criteria.

Example 4. *(ex. 3 continued) The following table augments the one presented in example 2 with the coalitions resulting from comparisons between the reference objects $A_1, A_2, B_1, B_2, C_1, C_2$ and the candidates Z_1, Z_2, Z_3.*

	$\star\star\star$		$\star\star$		$\star$		?	?	?
	A_1	A_2	B_1	B_2	C_1	C_2	Z_1	Z_2	Z_3
A_1	–	–	**abc**	**abd**	**abc**	**abd**	**N**	**N**	**N**
A_2	–	–	**N**	**abd**	**abc**	**abd**	**N**	acd	**N**
B_1	d	bd	–	–	**abd**	**abd**	**N**	ad	bcd
B_2	acd	ac	–	–	**abc**	**abd**	cd	acd	acd
C_1	d	d	acd	bd	–	–	acd	acd	cd
C_2	cd	c	c	bcd	–	–	cd	cd	cd
Z_1	d	b	(ab)	bd	(ab)	**abd**	–	–	–
Z_2	bd	bc	**abc**	bd	(ab)	**abd**	–	–	–
Z_3	(ab)	(ab)	(ab)	**abd**	(ab)	**abd**	–	–	–

Non-bracketed coalitions have already been sorted according to the preference information: boldfaced coalitions are those previously established as sufficient, the others are insufficient. Bracketed coalitions are yet undecided. $\forall z \in \{Z_1, Z_2, Z_3\}$, $\overrightarrow{o}(z, \mathcal{P})$ *appears in the corresponding line, and* $\overleftarrow{o}(z, \mathcal{P})$ *in the appropriate column.*

In this section, we specify the mapping between these observations and the output of the classifier system, the *recommendation* $K(z, \mathcal{P}) \subset \widehat{C}$ and an *explanation* $\mathcal{E}(k, \mathcal{P})$ supporting it.

4.1 Possible assignments

As defined by the works of [21] about *necessary* and *possible* preference relations, the definition of *possible assignments* is closely related to the notion of *consistency* of an assignment with respect to the corpus of preference information. Defining, as we did in Section 3, $\Phi(\mathcal{P})$ as the set of preference parameters compatible to $\mathcal{P}$, and assuming it is not empty:

- *necessary* assignments are yielded by *every* possible completion of these preference parameters;
- *possible* assignments are yielded by *some* possible completion of these preference parameters;
- *impossible* assignments are yielded by *no* possible completion of these preference parameters;

These sets of assignments are concisely described referring to the set:

$$C(z, \mathcal{P}) := \{k \in \widehat{C} : \Phi(\mathcal{P} \cup \{(z, k)\}) \neq \emptyset\}$$

A possible assignment is in $C(z, \mathcal{P})$, an impossible one is not. When $C(z, \mathcal{P})$ boils down to a singleton, then it is a necessary assignment for z.

This definition of *possible assignment* is straightforward to implement, simply iterating through the set of possible assignments classes $k \in \widehat{C}$, updating the preference information $\mathcal{P}' \leftarrow \mathcal{P} \cup \{(z, k)\}$, and checking the consistency of $\mathcal{P}'$. Unfortunately, it is a tricky notion when it comes to explaining. The actual unveiling of a Boolean judgment function compatible to the assignment is not very appealing, as it implies the assignment of $|\mathbb{B}^N|$ truth values, with some possible arbitrariness, and seems to defeat both the *no jargon* and the *no inference* principles. Consequently, we adopt the following principle:

Everything is possible, unless proven otherwise.

Doing so shifts the burden of proof towards impossibility, focusing on the exhibition of constraints restricting the set $C(z, \mathcal{P})$. We aim at *explaining* these constraints thanks to *statements* of the form $[premises : conclusions]_{scheme}$. We define several *argument schemes*, as formalized by [47] in order to capture stereotypical patterns of human reasoning used in Rhetoric for a long time [4, 5, 32], and widely used in Artificial Intelligence systems since [33, 35, 34, 44]. These schemes specify the nature and conditions imposed to both premises and conclusions, yielding to valid arguments. We are looking for *complete* explanations, so we must ensure the validity of the implication $premises \Rightarrow conclusions$, and provide *grounded* sets of statements, such that any premise is either the conclusion of another argument, or directly referencing the assumed available information (pairwise comparisons between the reference objects or the candidate, based on criteria or assignment).

In order to make apparent the cause of impossibility, we consider the potential consequences of assigning a candidate to a class through the *additional (in)sufficient coalitions conditional to the assignment of the candidate* z *to the class* k:

$$\mathbf{\Delta}\mathcal{T}_{\mathcal{P}}(z, k) := \mathcal{T}_{\mathcal{P} \cup \{(z,k)\}} \setminus \mathcal{T}_{\mathcal{P}};\ \mathbf{\Delta}\mathcal{F}_{\mathcal{P}}(z, k) := \mathcal{F}_{\mathcal{P} \cup \{(z,k)\}} \setminus \mathcal{F}_{\mathcal{P}}$$

We rewrite the impossibility of assigning the candidate z to the class k using the *conflicting constraints* characterization of inconsistency. We consider four potential sources of impossibility, sorted by evidence: $C(z, \mathcal{P}) = \bigcap_{i \in \{1,2,3\}} K_i(z, \mathcal{P})$ where:

- $K_1(z, \mathcal{P}) := \{k \in \widehat{C} : \mathcal{T}_{\mathcal{P}} \cap \Delta\mathcal{F}_{\mathcal{P}}(z, k) = \emptyset\}$ highlights conflicts between established sufficient coalitions, and the assignment of z;
- $K_2(z, \mathcal{P}) := \{k \in \widehat{C} : \Delta\mathcal{T}_{\mathcal{P}}(z, k) \cap \mathcal{F}_{\mathcal{P}} = \emptyset\}$ highlights conflicts between established insufficient coalitions, and the assignment of z;
- $K_3(z, \mathcal{P}) := \{k \in \widehat{C} : \Delta\mathcal{T}_{\mathcal{P}}(z, k) \cap \Delta\mathcal{F}_{\mathcal{P}}(z, k) = \emptyset\}$ takes into account the least obvious situation where some assignment of z may be self-contradictory, without conflicting with any previously acknowledged information.

The next section details the impossibilities captured by the set $K_1(z, \mathcal{P})$, and proposes a supporting explanation $\mathcal{E}_1(z, \mathcal{P})$, while the two other cases are briefly presented in section 4.3. Section 4.4 briefly addresses implementation details, and section 4.5 raises and discusses some issues with the model.

4.2 Assignments contradicting previously established sufficient coalitions

In this section, we focus on the set $K_1(z, \mathcal{P}) := \{k \in \widehat{C} : \mathcal{T}_{\mathcal{P}} \cap \Delta\mathcal{F}_{\mathcal{P}}(z, k) = \emptyset\}$. As seen in the previous section this set provides a range of possible assignments for the candidate z, and partially implements the model described by the manifesto exposed in section 2. We first describe $K_1(z, \mathcal{P})$ as an intersection of constraints, for which we provide a description based on arguments. We prove $K_1(z, \mathcal{P})$ is an interval of $\widehat{C}$, and provide a short, yet complete, explanation accounting for this recommendation.

By construction, the recommended set $K_1(z, \mathcal{P})$ is built in order to reject some impossible assignments. To illustrate and come to understand its behavior, we fabricate a situation that specifically trigger this rejection flag. Suppose we know that:

(1) the coalition of criteria $T \in \mathbb{B}$ is already known to be sufficient, and
(2) the candidate $z \in \mathbb{X}$ is at least as good as the reference object $\underline{x}^\star \in \mathbb{X}^\star$, assigned to class $\underline{k} \in \widehat{C}$, for every criteria in T.

Then, z outranks $\underline{x}^\star$ and cannot be assigned a class strictly worse than $\underline{k}$ by application of Rule 1. This constraint is captured by the set $K_1(z, \mathcal{P})$, as the assignment of z to any class $k \prec \underline{k}$ would lead to conclude that the coalition of criteria $O_N(z, \underline{x}^\star)$ is insufficient, so that the coalition of criteria T would belong to both sets $\Delta\mathcal{F}_{\mathcal{P}}(z, k)$ and $\mathcal{T}_{\mathcal{P}}$. Consequently, $k \notin K_1(z, \mathcal{P})$
If we replace the assumption (2) by:

(2') the reference object $\overline{x}^\star \in \mathbb{X}^\star$, assigned to class $\overline{k} \in \widehat{C}$, is at least as good as the candidate $z \in \mathbb{X}$ for every criteria in T.

then $\overline{x}^\star \in \mathbb{X}^\star$ outranks z and z cannot be assigned to a class strictly better than $\overline{k}$, as

$$k \succ \overline{k} \Rightarrow \mathcal{T}_{\mathcal{P}} \ni T \subseteq O_N(\overline{x}^\star, z) \in \Delta\mathcal{F}_{\mathcal{P}}(z, k) \Rightarrow k \notin K_1(z, \mathcal{P})$$

Reciprocally, any assignment $k_0 \notin K_1(z, \mathcal{P})$, results in a non-empty intersection $\mathcal{T}_{\mathcal{P}} \cap \Delta\mathcal{F}_{\mathcal{P}}(z, k_0)$, which involves at least one sufficient coalition $T \in \mathcal{T}_{\mathcal{P}}$, as in assumption (1), and one stronger, insufficient coalition resulting either from the observations $\overrightarrow{o}(z, \mathcal{P})$, as in assumption (2), or from $\overleftarrow{o}(z, \mathcal{P})$, as in (2').

A statement of type (1) needs to be backed by evidence, so we introduce two argument schemes:

Definition 1. *For any reference objects $a^\star, b^\star \in \mathbb{X}^\star$ and any coalition of criteria $T \in \mathbb{B}^N$, we say the tuple $[a^\star, b^\star : T]_\mathcal{T}$ instantiates the argument scheme* ESTABLISHED SUFFICIENT COALITION$(\mathcal{P})$ *if, and only if, $a^\star \succ_\mathcal{P} b^\star$ and $\forall i \in N \setminus T,\ b_i^\star > a_i^\star$. We also say the tuple $[\emptyset : N]_1$ instantiates the argument scheme* WEAK DOMINANCE.

Proposition 2 (Argumentative structure of the sufficient coalitions)**.**

$$\mathcal{T}_\mathcal{P} = \{N\} \cup \bigcup_{[a^\star, b^\star : T]_\mathcal{T}} \{T\}$$

The sufficient coalitions are exactly the conclusions of the arguments instantiating the ESTABLISHED SUFFICIENT COALITION$(\mathcal{P})$ *scheme.*

In order to account for the atoms of reasoning (2) and (2') and present them to the recipient of the recommendation, we define the corresponding argument schemes.

Definition 2. *For any coalition of criteria $T \in \mathbb{B}^N$, any reference object $x^\star \in \mathbb{X}^\star$ and any class $c \in \widehat{C}$, we say*

- *the tuple $[T, x^\star : c]_{\mathcal{T}/\overrightarrow{\sigma}}$ instantiates the argument scheme* CANDIDATE OUTRANKING$(z, \mathcal{P})$ *if, and only if, $T \in \mathcal{T}_\mathcal{P}$ and $\forall i \in T, z_i \geq x_i^\star$ and $class(x^\star) = c$.*
- *the tuple $[T, x^\star : c]_{\mathcal{T}/\overleftarrow{\sigma}}$ instantiates the argument scheme* CANDIDATE OUTRANKED$(z, \mathcal{P})$ *if, and only if, $T \in \mathcal{T}_\mathcal{P}$ and $\forall i \in T, x_i^\star \geq z_i$ and $class(x^\star) = c$*

Proposition 3 (Argumentative structure of the recommendation)**.**

$$K_1(z, p) = \widehat{C} \cap \bigcap_{[T, \underline{x}^\star : \underline{k}]_{\mathcal{T}/\overrightarrow{\sigma}}} \{k \succsim \underline{k}\} \cap \bigcap_{[T, \overline{x}^\star : \overline{k}]_{\mathcal{T}/\overleftarrow{\sigma}}} \{k \precsim \overline{k}\}$$

Proposition 3 is a concise rewording of the necessary and sufficient conditions for a given class *not* to belong to the set $K_1(z, \mathcal{P})$ detailed previously. As a corollary, it shows that $K_1(z, \mathcal{P})$ is an interval of $\widehat{C}$. Consequently, $K_1(z, \mathcal{P})$ can be completely described by a pair $(\underline{k_1}, \overline{k_1})$ such that:

- $K_1(z, \mathcal{P}) = [\underline{k_1}, \overline{k_1}]$;
- the lower bound $\underline{k_1}$ is *maximal*, as there is no class strictly better than $\underline{k_1}$ which is supported by an argument instantiating the CANDIDATE OUTRANKING$(z, \mathcal{P})$ scheme. It is *trivial* if $\underline{k_1} = \min \widehat{C}$ (either when the set CANDIDATE OUTRANKING$(z, \overline{\mathcal{P}})$ is empty, or when it does not support a stronger outcome), in which case it does not need any explanation. If $\underline{k_1} \succ \min \widehat{C}$, then it admits at least one *explanation* $\underline{E_1}$ composed of an argument $[T, \underline{x}^\star : \underline{k_1}]_{\mathcal{T}/\overrightarrow{\sigma}} \in$ CANDIDATE OUTRANKING backed by an argument $[a^\star, b^\star : T]_\mathcal{T} \in$ ESTABLISHED SUFFICIENT COALITION;
- the upper bound $\overline{k_1}$ is *minimal*, as there is no class strictly worse than $\overline{k_1}$ which is supported by an argument instantiating the CANDIDATE OUTRANKED$(z, \mathcal{P})$ scheme. It is *trivial* if $\overline{k_1} = \max \widehat{C}$, in which case it does not need any explanation. If $\underline{k_1} \prec \max \widehat{C}$, then it admits at least one *explanation* $\overline{E_1}$ composed of an argument $[T', \overline{x}^\star : \overline{k_1}]_{\mathcal{T}/\overleftarrow{\sigma}} \in$ CANDIDATE OUTRANKED backed by an argument $[a^\star, b^\star : T']_\mathcal{T} \in$ ESTABLISHED SUFFICIENT COALITION.

Finally, the recommended interval $K_1(z, \mathcal{P})$ is supported by an explanation $\mathcal{E}_1$ in the form of a pair $(\underline{E_1}, \overline{E_1})$, where $\underline{E_1}$ and $\overline{E_1}$ can be either the empty set or a pair of arguments. Taken together, all these 0, 2 or 4 arguments prove that any assignment $k \in \widehat{C} \setminus K_1(z, \mathcal{P})$ should be rejected as "impossible". Such explanation is not necessarily unique, and we denote by $\widehat{\mathcal{E}_1}(z, \mathcal{P})$ the set of suitable explanations.

Example 5. *(ex. 4 continued)*

Using the table presented in Example 4, the set K_1 can be interpreted as "a candidate cannot be assigned a class laying strictly on the right of, nor a class strictly above, a case containing a boldfaced coalition": Consequently,

- $\begin{cases} K_1(Z_1, \mathcal{P}_{ex}) = \{\star, \star\star\} \\ \mathcal{E}_1(Z_1, \mathcal{P}_{ex}) \ni (\emptyset, \{[\emptyset : N]_1, [N, B_1 : \star\star]_{\mathcal{T}/\overleftarrow{\sigma}}\}) \end{cases}$
  as B_1 is rated $\star\star$ and dominates Z_1.
- $\begin{cases} K_1(Z_2, \mathcal{P}_{ex}) = \{\star\star, \star\star\star\} \\ \widehat{\mathcal{E}_1}(Z_2, \mathcal{P}_{ex}) \ni (\{[A_1, C_1 : \texttt{abc}]_\mathcal{T}, [\texttt{abc}, B_1 : \star\star]_{\mathcal{T}/\overrightarrow{\sigma}}\}, \emptyset) \end{cases}$
  as C_1 is better than A_1 on criterion $\texttt{d}$ while ranked strictly below (establishing $N \setminus \texttt{d} = \texttt{abc}$ as a sufficient coalition of criteria), and Z_2 is at least as good as B_1 on every criteria except $\texttt{d}$, so Z_2 is not ranked lower than B_1's $\star\star$
- $\begin{cases} K_1(Z_3, \mathcal{P}_{ex}) = \{\star\star, \star\star\star\} \\ \widehat{\mathcal{E}_1}(Z_2, \mathcal{P}_{ex}) \ni (\{[A_2, C_2 : \texttt{abd}]_\mathcal{P}, [\texttt{abd}, B_2 : \star\star]_{\mathcal{T}/\overrightarrow{\sigma}}\}, \emptyset) \end{cases}$
  as $\texttt{abd}$ is a sufficient coalition of criteria (because, e.g. A_2 is ranked $\star\star\star$ and C_2 is $\star$, while C_2 bests A_2 on criterion $\texttt{c}$), and $B_2(\star\star)$ is not worse than Z_2 on criteria $\texttt{abd}$, so it cannot be ranked strictly worse than Z_2.

4.3 Other impossible assignments

The set $K_2(z, \mathcal{P})$ is defined symmetrically from $K_1(z, \mathcal{P})$ with respect to sufficient and insufficient coalitions. Assignments *not* in $K_2(z, \mathcal{P})$ result from the collision of a coalition of criteria known to be insufficient, and the observation of a candidate object resulting in an even weaker coalition, so outranking is excluded, and all the classes strictly above or below (depending on the direction of observation) the one of the reference object are therefore forbidden. *Mutadis mutandis*, we can define the argument schemes ESTABLISHED INSUFFICIENT COALITION$(\mathcal{P})$, WEAKLY DOMINATED, CANDIDATE NOT OUTRANKING$(z, \mathcal{P})$, CANDIDATE NOT OUTRANKED$(z, \mathcal{P})$ and obtain the same structural results, leading to define similar explanations for the lower and upper bounds of the interval $K_2(z, \mathcal{P})$.

Example 6. *(ex. 4 continued)*

Using the table presented in Example 4, the set K_2 interprets the insufficient coalitions of the table, those not boldfaced nor parenthesized. A candidate cannot be assigned a class strictly below, nor strictly on the left, of such cases. For instance, $O_N(B_2, Z_1) = \texttt{cd} \in \mathcal{T}_\mathcal{P}$ (because $O_N(C_2 \prec_\mathcal{P} A_1) = \texttt{cd}$, for instance), so Z_1 is not outranked by B_2 and should be at least assigned the same class ($\star\star$), and $O_N(Z_1, B_2) = \texttt{bd} \in \mathcal{T}_\mathcal{P}$ (for instance because it is weaker than $\texttt{bcd} = O_N(C_2 \prec_\mathcal{P} B_2)$), so Z_1 does not outrank B_2 and should not be assigned a strictly better class ($\star\star$). Finally, $K_2(Z_1, \mathcal{P}) = \{\star\star\}$.

The set $K_3(z, \mathcal{P})$ has a totally different structure. The contradictions that it catches are counter-factual by nature. They exclude assignments of classes that would entail inconsistent judgments on yet undecided coalitions of criteria. In particular, such reasoning does not guarantee that $K_3(z, \mathcal{P})$ is an interval, as some class may very well be excluded while its neighboring classes are not. In this paper, we do not propose any argumentative support for this set.

Example 7. *(ex. 4 continued)*

$\star\star \notin K_3(Z_3, \mathcal{P})$, as assigning Z_3 to class $\star\star$ would lead to a conundrum concerning the yet undecided coalition $\mathtt{ab}$, which should simultaneously be considered as sufficient (as Z_3 would outrank C_1 and $O_N(Z_3, C_1) = \mathtt{ab}$) and insufficient (as Z_3 would be outranked by A_1 and $O_N(Z_3, A_1) = \mathtt{ab}$).

4.4 Implementation

Depending on the number of candidates examined through the same preference information, and the respective cardinalities of the sets of criteria $|N|$ and reference objects $|\mathbb{X}^\star|$, we propose 3 different implementation strategies:

Direct confrontation of arguments. Recommendation and explanation is $\mathcal{O}(|\mathbb{X}^\star|^3)$, checking model consistency is $\mathcal{O}(|\mathbb{X}^\star|^3)$

Storage of observations. Intermediate computation of $O_N(\prec_\mathcal{P})$ and $O_N(\succ_\mathcal{P})$ takes $\mathcal{O}(|\mathbb{X}^\star|^2)$ time and leads to recommendation and explanation in $\mathcal{O}(|\mathbb{X}^\star| \times 2^{|N|})$ and consistency in $\mathcal{O}(2^{2|N|})$

Full description of the coalitions of criteria. Computing the sets $\mathcal{T}_\mathcal{P}$ and $\mathcal{F}_\mathcal{P}$ demands to traverse the hypercube $\mathbb{B}^N$, takes $\mathcal{O}(|\mathbb{X}^\star|^2 \times |N| 2^{|N|-1})$ time, demands much more memory, but simultaneously decides consistency and outputs recommendations and explanations in $\mathcal{O}(|\mathbb{X}^\star|)$ time.

4.5 No recommendation

A perfect recommendation would consist in a singleton, the only possible assignment for the candidate, which would appear as *necessary*. At the other end of the informative spectrum, a recommendation $K(z, \mathcal{P}) \equiv \widehat{C}$ does not deliver a single bit of information. There are also outliers, bizarre situations, e.g.

- the recommended assignments do not form an interval - there are holes in it! For this reason, we would consider $K(z, \mathcal{P}) := K_1(z, \mathcal{P})$ or $K(z, \mathcal{P}) := K_2(z, \mathcal{P})$ or $K(z, \mathcal{P}) := K_1(z, \mathcal{P}) \cap K_2(z, \mathcal{P})$, or at least take the convex hull of $C(z, \mathcal{P})$ to avoid the peculiarities coming from dealing with yet undecided coalitions of criteria dealt with $K_3(z, \mathcal{P})$;
- there is no possible assignment! There is no guarantee whatsoever that sets $K_1(z, \mathcal{P})$ and $K_2(z, \mathcal{P})$ are not disjoint, and, more fundamentally, while we have proven both sets have an interval structure, they may very well be *empty* intervals, when their lower bound exceed their upper bound. For instance, consider the following situation:

Example 8. *Objects are evaluated according to three criteria $\mathtt{a}, \mathtt{b}, \mathtt{c}$ valued on the same scale $\{A > B > C\}$; there are four reference objects $\mathbb{X}^\star := \{A_1, A_2, B_1, B_2\}$ assigned to two classes $\widehat{C} := \{\star \prec \star\star\}$ and a candidate Z:*

Object	$\mathtt{a}$	$\mathtt{b}$	$\mathtt{c}$	*Assignment*
A_1	A	B	B	$\star\star$
A_2	B	A	B	$\star\star$
B_1	C	C	A	$\star$
B_2	C	A	C	$\star$
Z	B	C	B	?

Observations are detailed in the following table:

	$\star\star$		$\star$		?
	A_1	A_2	B_1	B_2	Z
A_1	–	–	**ab**	**ac**	**ab**
A_2	–	–	**ab**	**bc**	**ab**
B_1	$\mathtt{c}$	$\mathtt{c}$	–	–	**bc**
B_2	$\mathtt{b}$	$\mathtt{b}$	–	–	$\mathtt{b}$
Z	$\mathtt{c}$	**ac**	**ab**	**ac**	–

The set $K_1(z, \mathcal{P}) = [\star\star, \star]$ is empty, with the formal explanation $(\{[A_1, B_2 : \mathtt{ac}]_\mathcal{P}, [\mathtt{ac}, A_2 : \star\star]_{\mathcal{T}/\overrightarrow{\sigma}}\}, \{[A_2, B_2 : \mathtt{bc}]_\mathcal{T}, [\mathtt{bc}, B_1 : \star]_{\mathcal{T}/\overleftarrow{\sigma}}\}) \in \widehat{\mathcal{E}}_1(z, \mathcal{P})$. Z outranks A_2 so it cannot be assigned less than $\star\star$, whereas it is outranked by B_2, so it cannot be assigned more than $\star$.

Example 8 depicts a disturbing situation concerning the *necessary outranking relation* $\mathcal{N}_\mathcal{P} := \bigcap_{\phi \in \Phi(\mathcal{P})} S_\phi$, as we have found a triplet $A_2, B_1, Z \in \mathbb{X}$ such that $A_2 \mathcal{N}_\mathcal{P} B_1 \mathcal{N}_\mathcal{P} Z \mathcal{N}_\mathcal{P} A_2$. This intransitivity of preference is a particular instance of the *Condorcet paradox* [14], known to occur in majority voting. It is also a well known issue for the outranking-based MCDA methods, such as the ELECTRE family, that rely on a qualified, weighted majority vote to aggregate the comparisons of alternative on each criterion, and has been identified as a potential issue in the case of a necessary ELECTRE model [20], and is also well detailed in the seminal article [24] introducing the principle of classification without frontiers based on weighted majority voting. Example 8 shows it definitely occurs in the case where the outranking relation is built from non-compensatory rules applied to an observed classification, without any assumption made concerning the inter-criteria aggregation procedure. Unfortunately, when such a cycle involves two reference objects in different classes (here, $A_2 \succ_\mathcal{P} B_1$), the decision aiding model fails to deliver any meaningful recommendation. More disturbingly, it does so

- without being itself inconsistent: the model is totally capable of accounting for the learning set;
- with a solid explanation for why nothing is possible.

This unfortunate situation is, alas, not confined to carefully crafted toy examples, as these Condorcet paradoxes are linked to Arrow's impossibility theorem [6], so that we can expect to always encounter some problematic situations, unless one of the criterion is a *dictator* (i.e. only the coalition of criteria containing it are sufficient), which means the situation does not need to be addressed by a multi-criteria method to begin with.

5 Conclusion

We have reformulated an implied feature of MCDA, the assumed *interpretability* of the various models, as an explicit objective, the *accountability* of the process, and a tangible output, an *explanation* supporting the recommendation. In the case of a sorting problem, where the classes are ordered but are not separated by a frontier, we propose a set of *argument schemes* implementing the explanation.

Disappointingly, but unsurprisingly, we uncover some dysfunctional features of the model, that make it *in fine* hardly suitable as a reliable artificial recommender system. Nevertheless, we believe that *accountability* is a novel direction worth pursuing for MCDA systems, and that the opportunity of explaining recommendations by implementing specific argument schemes could be extended to other, established, MCDA models.

This model is inadequate, as it sometimes offer non-viable recommendations. An obvious venue is to consider relaxing the very stiffening design constraints. In particular, reestablishing the frontiers between classes leads to the MR-Sort and NCS models (without veto), known to eschew the Condorcet paradox, which have recently known some development [40, 41]. Also, argumentation technology offers another possible venue, as it is increasingly being recognized, in fields of Artificial Intelligence like multi agent systems, for its usefulness to refine the reasoning capabilities of artificial agents [34, 44]. In

such a framework, the built-in defeasible nature of arguments could make it possible to weighs the different sources of impossibility and select a "sufficiently justified" recommendation. Abstract argumentation has started to be associated to MCDA by [2, 3], while dialogue techniques have been introduced by [31, 30, 26];

Also, if dysfunctional models maybe bad in the role of reliably providing good recommendation, they still have value, as they could capture real world (flawed) reasoning. For instance, in a decision situation where the decision maker has to defend her position to some stakeholders, it does not seem totally unlikely that a potential *adversary* of the defended recommendation would argue along the lines of the model presented here. The role of such adversarial models is investigated in [42].

REFERENCES

[1] I. Alvarez, 'Explaining the result of a Decision Tree to the End-User', in *Proceedings of the 16th ECAI*, pp. 411–415, (2004).

[2] L. Amgoud, J-F Bonnefon, and H. Prade, 'An argumentation-based approach to multiple criteria decision', in *ECSQARU*, pp. 269–280, (2005).

[3] L. Amgoud and H. Prade, 'Using arguments for making and explaining decisions', *Artificial Intelligence*, **173**(3), 413–436, (2009).

[4] Aristotle, *On Sophistical Refutations*, Harvard University Press, 1928.

[5] Aristotle, *Topics*, Harvard University Press, 1939.

[6] Kenneth J Arrow, 'A difficulty in the concept of social welfare', *The Journal of Political Economy*, 328–346, (1950).

[7] K. Belahcene, C. Labreuche, N. Maudet, V. Mousseau, and W. Ouerdane, 'Explaining robust additive utility models by sequences of preference swaps', *Theory and Decision*, (2016).

[8] V. Belton and T. Stewart, *Muliple Criteria Decision Analysis: An Integrated Approach*, Kluwer Academic, 2002.

[9] E. Boros, Y. Crama, P. Hammer, T. Ibaraki, A. Kogan, and K. Makino, 'Logical analysis of data: classification with justification', *Annals of Operations Research*, **188**(1), 33–61, (2011).

[10] G. Bous, Ph. Fortemps, F. Glineur, and M. Pirlot, 'Acuta: A novel method for eliciting additive value functions on the basis of holistic preference statements', *EJOR*, **206**(2), 435–444, (2010).

[11] D. Bouyssou and T. Marchant, 'An axiomatic approach to noncompensatory sorting methods in mcdm, i: The case of two categories', *EJOR*, **178**(1), 217–245, (2007).

[12] Denis Bouyssou, 'Some remarks on the notion of compensation in mcdm', *EJOR*, **26**(1), 150–160, (1986).

[13] O. Cailloux, *Indirect elicitation of multicriteria sorting models*, Ph.D. dissertation, Ecole Centrale Paris, 2012.

[14] Condorcet, *Essai sur l'application de l'analyse a la probabilite des decisions rendues a la pluralite des voix [microform] / par M. le Marquis de Condorcet*, Imprimerie royale Paris, 1785.

[15] Yves Crama, Peter L Hammer, and Toshihide Ibaraki, 'Cause-effect relationships and partially defined boolean functions', *Annals of Operations Research*, **16**(1), 299–325, (1988).

[16] J.M. Devaud, G. Groussaud, and E. Jacquet-Lagreze, 'UTADIS: Une methode de construction de fonctions d'utilite additives rendant compte de jugements globaux', in *European working group on MCDA, Bochum , Germany*, (1980).

[17] L. Dias, V. Mousseau, J. Figueira, and J. Clımaco, 'An aggregation/disaggregation approach to obtain robust conclusions with electre tri', *EJOR*, **138**(2), 332–348, (2002).

[18] Peter C. Fishburn, 'Noncompensatory preferences', *Synthese*, **33**(2/4), 393–403, (1976).

[19] S. Greco, M. Kadziński, and R. SŁowiński, 'Selection of a representative value function in robust multiple criteria sorting', *Computers & Operations Research*, **38**(11), 1620–1637, (2011).

[20] S. Greco, M. Kadzinskiand V. Mousseau, and R. Slowinsk, 'Electre^{GKMS}: Robust ordinal regression for outranking methods', *EJOR*, **214**(1), 118–135, (2011).

[21] S. Greco, R. Słowinski, J. Figueira, and V. Mousseau, 'Robust ordinal regression', in *Trends in Multiple Criteria Decision Analysis*, 241–284, Springer Verlag, (2010).

[22] Salvatore Greco, Vincent Mousseau, and Roman Słowiński, 'Ordinal regression revisited: multiple criteria ranking using a set of additive value functions', *EJOR*, **191**(2), 416–436, (2008).

[23] R.L. Keeney and H. Raiffa, *Decisions with multiple objectives: Preferences and value tradeoffs*, J. Wiley, New York, 1976.

[24] M. Köksalan, V. Mousseau, O. Ozpeynirci, and S. Bilgin Ozpeynirci, 'An outranking-based approach for assigning alternatives to ordered classes', *Naval Research Logistics*, **56**(1), 74–85, (February 2009).

[25] M. Köksalan and C. Ulu, 'An interactive approach for placing alternatives in preference classes', *EJOR*, **144**(2), 429–439, (2003).

[26] C. Labreuche, N. Maudet, W. Ouerdane, and S. Parsons, 'A dialogue game for recommendation with adaptive preference models', in *Proceedings AAMAS*, pp. 959–967, (2015).

[27] Christophe Labreuche, Nicolas Maudet, and Wassila Ouerdane, 'Justifying Dominating Options when Preferential Information is Incomplete.', in *ECAI'12*, volume 242, (2012).

[28] O. I. Larichev and H. M. Moshkovich, *The Method ORCLASS for Ordinal Classification of Multiattribute Alternatives*, 189–237, 1997.

[29] A. Leroy, V. Mousseau, and M. Pirlot, 'Learning the parameters of a multiple criteria sorting method', in *ADT*, pp. 219–233. Springer, (2011).

[30] W. Ouerdane, N. Maudet, and A. Tsoukias, 'Argumentation theory and decision aiding', in *Trends in Multiple Criteria Decision Analysis*, 177–208, Springer, (2010).

[31] W. Ouerdane, Nicolas Maudet, and Alexis Tsoukiàs, 'Argument schemes and critical questions for decision aiding process', in *COMMA*, pp. 285–296. IOS Press, (2008).

[32] C. Perelman and L. Olbrechts-Tyteca, *The new Rhetoric: A treatise on argumentation*, University of Notre Dame Press, 1969.

[33] J.L. Pollock, *Congnitive Carpentry : A Blueprint for how to Buid a Person*, Cambridge (Massachusetts), 1995.

[34] C. Reed and T.J. Norman, *Argumentation Machines : New Frontiers in Argument and Computation*, Kluwer Academic Publishers, 2003.

[35] C. Reed and D. Walton, 'Application of argumentation schemes', in *OSSA2001*, (2001).

[36] B. Roy, 'The outranking approach and the foundations of electre methods', *Theory and decision*, **31**(1), 49–73, (1991).

[37] B. Roy and D. Bouyssou, *Aide Multicritère à la Décision : Méthodes et Cas*, Economica, Paris, 1993.

[38] Ahti A Salo and Raimo P Hamalainen, 'Preference ratios in multiattribute evaluation (prime)-elicitation and decision procedures under incomplete information', *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, **31**(6), 533–545, (2001).

[39] Y. Siskos, E. Grigoroudis, and N. F. Matsatsinis, 'Uta methods', in *Multiple criteria decision analysis: State of the art surveys*, 297–334, (2005).

[40] O. Sobrie, V. Mousseau, and M. Pirlot, 'Learning a majority rule model from large sets of assignment examples', in *International Conference on ADT*, pp. 336–350, (2013).

[41] O. Sobrie, V. Mousseau, and M. Pirlot, 'Learning the parameters of a non compensatory sorting model', in *International Conference on ADT*, pp. 153–170, (2015).

[42] Ran Spiegler, 'Equilibrium in justifiable strategies: A model of reason-based choice in extensive-form games', *The Review of Economic Studies*, **69**(3), 691–706, (2002).

[43] N. Tintarev, 'Explanations of recommendations', in *Proc. ACM conference on Recommender systems*, pp. 203–206, (2007).

[44] B. Verheij, 'Artificial argument assistants for defeasible argumentation', *Artificial Intelligence*, **150**(1-2), 291–324, (2003).

[45] Ph. Vincke, *L'Aide Multicritère à la Décision*, Editions de l'Université de Bruxelles - Editions Ellispses, 1989.

[46] Philippe Vincke, 'Robust solutions and methods in decision-aid', *Journal of multicriteria decision analysis*, **8**(3), 181, (1999).

[47] D. Walton, *Argumentation schemes for Presumptive Reasoning*, Mahwah, N. J.,Erlbaum, 1996.

[48] W. Yu, *Aide multicritère à la décision dans le cadre de la problématique du tri: méthodes et applications*, Ph.D. dissertation, LAMSADE, Université Paris Dauphine, Paris, 1992.

Query-based learning of acyclic conditional preference networks from noisy data

Fabien Labernia and **Florian Yger** and **Brice Mayag** and **Jamal Atif** [1]

Abstract. Conditional preference networks (CP-nets) provide a powerful, compact, and intuitive graphical tool to represent the preferences of a user. However learning such a structure is known to be a difficult problem due to its combinatorial nature. We propose in this paper a new, efficient, and robust query-based learning algorithm for acyclic CP-nets. In particular, our algorithm takes into account the incoherences in the user's preferences or in noisy data by searching in a principled way the variables that condition the other ones. We provide complexity results of the algorithm, and demonstrate its efficiency through an empirical evaluation on synthetic and on real datasets.

1 Introduction

Representing, learning and reasoning over users preferences is a central question in many Artificial Intelligence related fields, and beyond. A large body of research has focused on preference representation and reasoning (e.g. [6, 11, 14, 21, 22, 23]), but few works have concerned their learning (e.g. [7, 8, 9, 12, 15, 16, 18]). This work focuses on **learning combinatorial preferences**, in the framework of conditional preference networks (CP-nets) as introduced in [5].

CP-nets are a formal framework for preference representation based on the notion of **ceteris paribus** (i.e "all other things being equal"). This notion of *ceteris paribus* captures an intuitive idea: it is difficult to express one preference between two totally different objects[2]. Nevertheless, it is easier to express one preference between two almost identical ones. In this work, we consider that *ceteris paribus* means that two objects differ only by one attribute value. For instance, if two hotels differ by their price *ceteris paribus*, it is probably easier to chose one of them. CP-nets implement this notion by factorizing the preferences, leading to a compact graphical representation.

Learning CP-nets is known to be NP-Complete [1, 5, 7], even for acyclic ones. Despite this 'negative' result, some works have tackled this problem, e.g. regression-based learning [9, 10, 17], learning by reduction to 2-SAT [8], and learning by user queries [7, 13, 15].

In all these approaches, the preferences of the users are considered to be coherent ones. Some other works take into account of noisy preferences [18, 19]. A **noise**, also called **incoherence** or **inconsistency**, is a preference that does not correspond to the true one, i.e. "I prefer $\mathbf{a}$ than $\mathbf{b}$" whereas the true preference is "I prefer $\mathbf{b}$ than $\mathbf{a}$". This noise can appear at random, as explained in [4], instead of an adversarial noise [3] which is not studied here.

[1] Université Paris-Dauphine, PSL Research University, CNRS, UMR 7243, LAMSADE, 75016 PARIS, email: {fabien.labernia,brice.mayag,florian.yger,jamal.atif}@lamsade.dauphine.fr

[2] An object can be a hotel having as a set of attributes: the number of rooms, the price, etc.

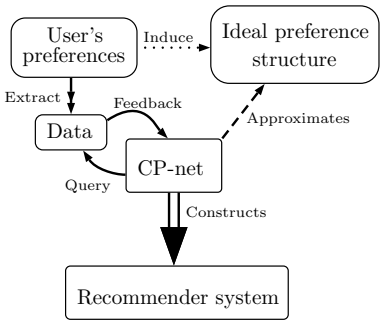


Figure 1. General scheme of a learning procedure for CP-nets.

In this work, we propose a new, efficient, and robust learning algorithm for CP-nets. Our aim is to come up with a learning procedure with the aim of recommendation systems as illustrated in Figure 1. Classically, a CP-net is constructed for each user. However, in this work, we use a CP-net as a way to aggregate preferences of many users. Our learning procedure can be seen as a way to learn an average CP-net of the common preferences between different users. To do this, we exploit the notion of **exact learning**, initially proposed by [2] in the context of query learning.

We focus on the **query learning** paradigm for CP-nets as introduced in [15]. More precisely, we proceed by querying (a set of users or a database) the preference between two objects that differ by just one attribute value), and in case of incoherence detection, we minimize its influence by choosing the optimal attributes that maximize the coherence of the overall learnt CP-net.

Our learning algorithm is composed of two phases: a general learning phase aiming at adding the preference in the graphical structure of the CP-net, and a parent search phase aiming at updating this graphical structure. This two-phases decomposition is classical in CP-nets learning. However, the originality of our approach lies in these both phases. In the general learning phase, our method add the less possible number of preferences in the structure which significantly decreases the computation time, and in the search parent phase, we propose a principle updating strategy that minimizes the incoherence of the overall structure by choosing the parent variable which splits the large number of rules stemming from the data.

2 Preliminaries

Let us first introduce some notations and concepts related to CP-nets [5].

2.1 Conditional preference networks (CP-nets)

Let $\mathbf{V} = \{X_1, \ldots, X_n\}$ be a set of n binary **variables** (variables are denoted by capital letters X and sets of variables are denoted by bold letters $\mathbf{X}$). Each variable $X \in \mathbf{V}$ is associated with a **domain** $Dom(\{X\}) = \{x, x'\}$ (to simplify the notation, we will omit the brackets and write $Dom(X)$) of values it can take. The value $x \in Dom(X)$ of a variable $X \in \mathbf{V}$ is called an **assignment**. We denote by $Dom(\mathbf{V}) = Dom(X_1) \times \ldots \times Dom(X_n)$ the domain of the values of $\mathbf{V}$. We call a **state** vector $\mathbf{x} \in Dom(\mathbf{X})$ which is an assignment of all $X_i \in \mathbf{X}$, with $\mathbf{X} \subseteq \mathbf{V}$ (if $\mathbf{X} = \mathbf{V}$, such a vector is called **outcome**).

We consider in this study a **strict preference relation** as a partial ordering $\succ$ on $Dom(\mathbf{V})$, i.e. $x \succ y$ meaning that an object x **is strictly preferred to** an object y.

Let $\mathbf{x} \in Dom(\mathbf{X})$ and $\mathbf{y} \in Dom(\mathbf{Y})$ be two states, with $\mathbf{X}, \mathbf{Y} \subseteq \mathbf{V}$, $\mathbf{X} \cap \mathbf{Y} = \emptyset$. The notation $\mathbf{xy} \in Dom(\mathbf{X} \cup \mathbf{Y})$ is the concatenation of the state $\mathbf{x}$ and the state $\mathbf{y}$.

Let $\mathbf{o}, \mathbf{o}' \in Dom(\mathbf{V})$ be two outcomes. We call a **swap**, denoted by $(\mathbf{o}, \mathbf{o}')_V$ (which induces $\mathbf{o} \succ \mathbf{o}'$), a pair of two outcomes such that the assignment of only one variable V changes between both. This variable is called the **swap variable** of $(\mathbf{o}, \mathbf{o}')_V$. Furthermore, $\mathbf{o}[\mathbf{X}]$ represents the vector of assignments of all variables in the set $\mathbf{X} \subseteq \mathbf{V}$ (when $\mathbf{X} = \mathbf{V}$, $\mathbf{o}[\mathbf{X}] = \mathbf{o}$).

A variable X is called a **parent variable** of another variable V if the assignment of X changes the preference over the assignments of V. More generally, $Pa(V)$ denotes the set of all parent variables of V. This defines a rule r, called **CP-rule**, of the form $r = (\mathbf{u} : v \succ v')$ with $V \in \mathbf{V}, Dom(V) = \{v, v'\}, \mathbf{U} = Pa(V) \subseteq \mathbf{V} \setminus \{V\}$, and $\mathbf{u} \in Dom(\mathbf{U})$. If $Pa(V) = \emptyset$, we just write $r = (v \succ v')$. We say that a CP-rule $r = (\mathbf{u} : v \succ v')$ **is linked by** V. These rules are stored in a structure called **CP-table** (for conditional preference table), which is unique for each variable $Y \in \mathbf{V}$ and is denoted by $CPT(Y)$. A CP-table contains the preferences over the assignment of Y for some states $\mathbf{x} \in Dom(Pa(Y))$. When all the possible states of $Pa(Y)$ are present, it is said to be **complete**. We note by $CPT(\mathbf{V})$ the union of all CP-tables in $\mathbf{V}$. More formally $CPT(\mathbf{V}) = \bigcup_{V \in \mathbf{V}} CPT(V)$.

Definition 1. *A **conditional preference network** (CP-net) $\mathcal{N} = (\mathbf{V}, A, CPT(\mathbf{V}))$ is a directed graph with $\mathbf{V}$ the set of vertices (representing the variables), A the set of directed arcs such that $(X, Y) \in A$ iff $X \in Pa(Y)$, and $CPT(\mathbf{V}) = \bigcup_{V \in \mathbf{V}} CPT(V)$. A CP-net is said **complete** if its associated CP-tables are complete. We call **separable CP-net** a CP-net $\mathcal{N}$ such that $A = \emptyset$, i.e. $Pa(V) = \emptyset$, $\forall V \in \mathbf{V}$.*

A graphic representation of a CP-net and the partial ordering of all possible outcomes is given in Figure 2. One can note that it is a complete and a non-separable CP-net which contains an independent variable (swimming pool) and a variable conditioned by another one (kitchen is conditioned by occupancy).

We say that a CP-net $\mathcal{N}$ is **cyclic** iff there exists a cycle in its associated graph. Otherwise, the CP-net is said **acyclic**. We restrict our work to acyclic CP-nets.

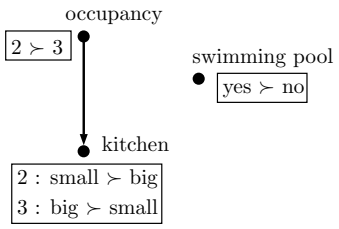


Figure 2. Complete CP-net with three variables.

2.2 Query learning algorithms for CP-nets

We discuss in this section the main query learning algorithms for CP-nets. We denote by n the number of variables of a CP-net, i.e. $|\mathbf{V}| = n$, by $\mathcal{N}_T$ the target CP-net, i.e. the structure that we want to fit, and by $\mathcal{N}_L$ our learned CP-net.

The first reported query learning algorithm for CP-nets in the literature [15] proceeds by asking different questions to a user in order to learn her preferences. The working assumption is that the user preferences are representable by a CP-net structure. Two types of questions can be distinguished:

- **equivalence queries** $\texttt{EQ}(\mathcal{N}_L, \mathcal{N}_T)$ which return `TRUE` if $\mathcal{N}_L$ is equivalent to $\mathcal{N}_T$[3], otherwise they return `FALSE` plus a counterexample in a form of a swap $(\mathbf{o}, \mathbf{o}')_V$ s.t. $\mathbf{o} \succ \mathbf{o}'$ representing a violated rule r.
- **membership queries** $\texttt{MQ}(\mathcal{N}_T, r)$, which return `TRUE` if the rule r is satisfied in $\mathcal{N}_T$ (denoted by $\mathcal{N}_T \models r$), otherwise they return `FALSE`.

When $\mathcal{N}_L \neq \mathcal{N}_T$, the equivalence query returns a swap counterexample $(\mathbf{o}, \mathbf{o}')_V$. This swap induces a rule r which (1) if it does not exist in $\mathcal{N}_L$, then we just have to add r in $\mathcal{N}_L$, or (2) if it is violated in $\mathcal{N}_L$, then we have to find a new parent variable for V. In [15], it has been shown that a linear number of equivalence queries ($O(|\mathbf{V}|)$) and a logarithmic number of membership queries ($O(\log_2 |\mathbf{V}|)$) are required to learn such a CP-net. For an in-depth explanation of the method, the interested reader can refer to [15].

The second algorithm [13] is neither based on *ceteris paribus* comparison nor equivalence and membership queries. It is an **online** algorithm that learns an **acyclic** CP-net and is decomposed into two phases:

- finding a separable CP-net by asking for each variable V the preference between v and v',
- updating the CP-table of each variable by finding the best set of parent variables using a set of confident[4] variables.

In this algorithm, all the parent variables are selected at the same time by phase 2. It seeks a subset of parent variables $\mathbf{P}$ from the set of all confident variables $\mathbf{C}$, i.e. $\mathbf{P} \subseteq \mathbf{C} \subseteq \mathbf{V}$. Moreover, for all $\mathbf{P} \subseteq \mathbf{C}$, the entire CP-table of the current variable V is created and tested until a good one is found (i.e. a CP-table that satisfies all the preferences). In the worst case, this algorithm needs 2^{2^n} operations to determine, for each variable, its parents and its corresponding CP-table.

[3] We say that two CP-nets $\mathcal{N}$ and $\mathcal{N}'$ are equivalent, denoted by $\mathcal{N} \equiv \mathcal{N}'$ iff they induce exactly the same preferences.

[4] We consider a variable V as **confident** if enough swap that induced rules of V are found.

Due to the exponential nature of this phase, it is necessary to limit the computation by bounding the size p of parent variables, the number e of edges in the target CP-net $\mathcal{N}_T$, and the number q of necessary swaps to conclude that a variable becomes confident. Finally, the algorithm can learn a CP-net in $O(n^p)$, with p the maximum number of parent variables. For more details, we refer the reader to [13].

3 Proposed algorithm for learning a CP-net

Let $r = (\mathbf{u} : v \succ v')$ be a rule with $V \in \mathbf{V}, Dom(V) = \{v, v'\}, Pa(V) = \mathbf{U} \subseteq \mathbf{V} \setminus \{V\}$ and $\mathbf{u} \in Dom(\mathbf{U})$. For the sake of clarity, we introduce in this section the notations $\bar{r} = (\mathbf{u} : v' \succ v)$ as the inverse rule of V, and $r^p = (\mathbf{u}p : v \succ v')$ as the augmentation of rule r with an assignment $p \in Dom(P)$ of a new parent variable $P \in \mathbf{V} \setminus (\mathbf{U} \cup \{V\})$.

As in the algorithm in [15], we query an oracle through $\texttt{EQ}(\mathcal{N}_L, \mathcal{N}_T)$ if our learned CP-net $\mathcal{N}_L$ is equivalent to its induced (the target) CP-net $\mathcal{N}_T$. This oracle can be either a user or a dataset that returns `TRUE` or `FALSE`. We suppose here that the oracle is able to answer, in a polynomial time, the following three queries:

1. the equivalence between two CP-nets (and returns a counterexample if not),
2. its preference between two outcomes in a swap,
3. another swap $(\mathbf{o}'', \mathbf{o}''')_V$ which respects some conditions (see Eq. (1) in Subsection 3.2).

We denote by Σ the oracle, and by $\mathcal{N}_T$ its proper CP-net. The target CP-net $\mathcal{N}_T$, cannot generally be explicitly known due to e.g. cognitive overloads for a user or the size of the database. However, we suppose that the oracle is able to differentiate its proper CP-net from the learned one.

Our algorithm, contrary to the state of the art approaches, tries to take into account incoherent (contradictory) preferences. Indeed, it may happen, for a user, to have **incoherent** preferences (she makes some mistakes or her preferences cannot be modeled by a CP-net). In case of databases, the data can be affected by noise, i.e. contradictory preferences. Hence, the learning procedure should be robust as much as possible to such a noise. In our approach, we introduce a **list of violated rules** L which cannot be represented in our CP-net either because we cannot add a parent to a variable to represent the rule, or even if such a parent exists, this rule cannot hold in $\mathcal{N}_L$. Then, we say that two CP-nets $\mathcal{N}_L$ and $\mathcal{N}_T$ are equivalent if the oracle compares these two CP-nets without using the rules contained in L, i.e. $\texttt{EQ}(\mathcal{N}_L, \Sigma, L)$.

Following previous learning algorithms, we decompose our procedure into a general learning phase, and a parent search phase. Still, our two phases are different from the ones in the state of the art.

3.1 General learning phase

In exact learning theory, we look for a perfect equivalence between the target and the learned structure. Following this perspective, we need to completely fit $\mathcal{N}_L$ and improve as much as possible the learning accuracy. Algorithm 1 corresponds to the general learning phase, starting with an empty CP-net $\mathcal{N}_L$.

$\mathcal{N}_L$ is updated by the rule r induced by the swap counterexample provided by the oracle. As in [5], two cases can occur: the inverse rule $\bar{r}$ is not present in $\mathcal{N}_L$, then we just have to add r to our CP-net. But if $\bar{r}$ is already present, then we must find a new parent to the variable V associated with the rule r. Since our Algorithm 1 does not always pick the real good parent (because of the presence of incoherences in the data), the parent search procedure can fail. If this happens (Line 11), we add r in the list of violated rules (i.e. a list that cannot be represented in $\mathcal{N}_L$). Otherwise (Line 7), we remove all the $CPT(V)$ and create a new one that contains r^p and $\bar{r}^{p'}$.

Algorithm 1: $\texttt{learningCPNet}(\Sigma, \mathcal{N}_T)$

Data: An oracle Σ which induces a target CP-net $\mathcal{N}_T$.
Result: A learned CP-net $\mathcal{N}_L$.

1 $L = \emptyset$;
2 $\mathcal{N}_L = (\mathbf{V}, A = \emptyset, CPT(\mathbf{V}) = \emptyset)$;
3 **while** *($\neg\texttt{EQ}(\mathcal{N}_L, \mathcal{N}_T, L)$)* **do**
4 Let $(\mathbf{o}, \mathbf{o}')_V \in \Sigma$ be a swap counterexample returned by `EQ` and $r = (\mathbf{u} : v \succ v')$ its induced rule with $\mathbf{U} = Pa(V)$;
5 **if** *($\bar{r} \in CPT(V)$)* **then**
6 $P \leftarrow \texttt{searchParent}((\mathbf{o}, \mathbf{o}')_V, \Sigma)$;
7 **if** *(P exists, with $Dom(P) = \{p, p'\}$)* **then**
8 $A \leftarrow A \cup \{(P, V)\}$;
9 $CPT(V) \leftarrow \{r^p, \bar{r}^{p'}\}$ where $p = \mathbf{o}[P]$;
10 $L \leftarrow L \setminus \{r'\},\ \forall r' \in L$ s.t. r' is linked by V;
11 **else** $L \leftarrow L \cup \{r\}$;
12 **else** $CPT(V) \leftarrow CPT(V) \cup \{r\}$;
13 **return** $\mathcal{N}_L$;

The algorithms in [13, 15] try to compute the complete CP-table of V and restart for each new parent variable, which leads to heavy computations. Nevertheless, we guess that in real world applications, CP-tables are not generally complete. Moreover, in case when the oracle refines its judgment, the preference in a swap can be different from a moment to the other, and an incoherence becomes coherent. Unfortunately, the opposite is also possible. However, we expect that its preferences becomes truthful as the time passes. Then it is not mandatory to compute the complete one in a greedy manner. This allows us to reduce the computational time. Besides, the missing entries are detected in our algorithm thanks to the equivalence queries.

3.2 Parent search phase

The parent search phase is the most important one in the learning procedure. This is due to the fact that several parent variables can be chosen, which may lead to bad decisions.

The procedure `searchParent` needs as an input the swap counterexample given by `EQ` and the oracle Σ. Its first step is to query the parent variable P of the swap $(\mathbf{o}, \mathbf{o}')_V$ (with $\mathbf{o} \succ \mathbf{o}'$) associated with variable V. The variable P must satisfy the following conditions:

1. It should preserve the assignment p between two comparable[5] outcomes. This is trivial in our case because we have a swap.
2. There should exist at least one other swap $(\mathbf{o}'', \mathbf{o}''')_V$ associated with the same variable V such that the preference on V is reversed and it contains the inverse assignment of P.

We can summarize these conditions in the following equation: let $(\mathbf{o}, \mathbf{o}')_V$ and $(\mathbf{o}'', \mathbf{o}''')_V$ two swaps, with $V, P \in \mathbf{V}$, which respect

$$\begin{aligned} &\mathbf{o}[V] = \mathbf{o}'''[V] \neq \mathbf{o}'[V] = \mathbf{o}''[V], \\ \text{and}\quad &\mathbf{o}[P] = \mathbf{o}'[P] \neq \mathbf{o}''[P] = \mathbf{o}'''[P]. \end{aligned} \tag{1}$$

[5] Two outcomes $\mathbf{o}$ and $\mathbf{o}'$ are **comparable** if either $\mathbf{o} \succ \mathbf{o}'$ or $\mathbf{o}' \succ \mathbf{o}$.

These constraints are modeled in Line 1. Since we restrict ourselves to acyclic CP-nets, a function `cycle` is used to test the acyclicity of $\mathcal{N}_L$ with the new parent variable.

We need to choose the good parent variable among all the available ones in $\mathbf{P}$. Instead of choosing a random $P \in \mathbf{P}$, we pick the variable P that minimizes the number of swaps that violate the rule induced by the current swap counterexample, i.e. let $(\mathbf{o}, \mathbf{o}')_V$ and $(\mathbf{o}'', \mathbf{o}''')_V$ be two swaps, with $V \in \mathbf{V}$ and $P \in Pa(V)$, then:

$$\begin{array}{rl} & (\mathbf{o}[P] = \mathbf{o}''[P] \text{ and } \mathbf{o}[V] \neq \mathbf{o}''[V]) \\ \text{or} & (\mathbf{o}[P] \neq \mathbf{o}''[P] \text{ and } \mathbf{o}[V] = \mathbf{o}''[V]). \end{array} \quad (2)$$

In case of equality, we randomly choose one.

Our parent search procedure is given in Algorithm 2. It is important to note that this algorithm can be parallelized using one process for each parent candidate, thanks to the independence of this search. However, this implementation is left for future work.

Algorithm 2: `searchParent`$((\mathbf{o}, \mathbf{o}')_V, \Sigma)$

Data: A swap $(\mathbf{o}, \mathbf{o}')_V$ and the oracle Σ.
Result: A parent variable P if there exists one, an error otherwise.

1 $\mathbf{P} \leftarrow \{P \in \mathbf{V} \setminus (\{V\} \cup Pa(V)) \mid \neg\texttt{cycle}(\mathcal{N} = (\mathbf{V}, A \cup \{(P, V)\}, CPT(\mathbf{V}))) \text{ and } \exists (\mathbf{o}'', \mathbf{o}''')_V \in \Sigma \text{ s.t. Eq. (1) returns } \texttt{TRUE}\}$;
2 **if** $(\mathbf{P} \neq \emptyset)$ **then**
3 $\quad$ **return** $\underset{P \in \mathbf{P}}{\operatorname{argmin}} \{\#(\mathbf{o}'', \mathbf{o}''')_V \in \Sigma \mid \text{Eq. (2) returns } \texttt{TRUE}\}$;
4 **else return** "parent not found";

Example 1. *Let us now try to learn a simple complete CP-net from Figure 2 (without the swimming pool variable). We begin by asking a couple if the empty learned CP-net is equivalent to their induced one. This is obviously not the case. Consider that they then give us the following swap counterexample* $((2,\ big), (3,\ big))_{occupancy}$*. The algorithm finds the rule* $r = (2 \succ 3)$ *which is added to* $\mathcal{N}_L$*. The next step is the equivalence query. Consider now that they answer the query by returning* $((2,\ small), (2,\ big))_{kitchen}$*. Then, the rule* $r = (small \succ big)$ *is deduced, and is added to* $\mathcal{N}_L$*. The process is repeated once again. Consider once more that the couple at this stage returns the following swap counterexample* $((3,\ big), (3,\ small))_{kitchen}$ *which induces the rule* $r = (big \succ small)$*. Since the inverse rule* $\bar{r} = (small \succ big)$ *already exists, the* `searchParent` *returns only the occupancy variable. Theses two new rules* $r^3 = (3 : big \succ small)$ *and* $\bar{r}^2 = (2 : small \succ big)$ *are then added in the CP-net.*

We end this subsection by giving some complexity results about these two algorithms.

Proposition 1. *Let* Σ *be an oracle. We define by* s *the number of swaps contained in* Σ*, i.e. holding in a database, or known by a user, and by* n *the number of variables in a CP-net. Algorithm 2 has a complexity of* $O(n^3 + ns)$*.*

Proof : Line 1 has to detect a cycle in $O(n+n^2) \approx O(n^2)$. The first n is the number of variables in the CP-net and the second n^2 corresponds to the maximum number of directed arcs in a directed graph. Finding a swap that respects the parent condition can be computed in $O(s)$. These steps have to be repeated for each parent candidate. Thus, Line 1 has a total complexity of $O(n(s+n^2) = O(n^3 + ns)$. Line 3 has to count the number of swaps in Σ that respect a given condition, it is in $O(ns)$. Hence, Algorithm 2 has a total complexity of $O(n^3 + ns)$. ■

Proposition 2. *Algorithm 1 has a complexity of* $O(2^p(n^4 + n^2 s + ne))$ *to compute* $\mathcal{N}_L$*, where* $p = \max_{V \in \mathbf{V}} \{|Pa(V)|\}$*,* e *is the time taken by* `EQ` *to return* `TRUE` *or* `FALSE`*, and* $\mathcal{N}_L$ *is an acyclic CP-net.*

Proof : We know from Prop. 1 that `searchParent` (Line 6) has a complexity of $O(n^3+ns)$. We now consider the `while` condition at Line 3. Suppose that $\mathcal{N}_T$ is a complete CP-net. Then, we must have complete CP-tables which imply, for p the max number of parents in $\mathcal{N}_T$, 2^p equivalence queries (one for each rule). Moreover, we have to remove all the rules in the CP-table when a new parent is found. In the worst case, we need $\sum_{i=1}^{p} 2^i = 2(2^p - 1)$ equivalence queries to learn one CP-table, so $2n(2^p - 1)$ equivalence queries in total. We finally have a complexity of $2n(2^p - 1)(n^3 + ns + e) \in O(2^p(n^4 + n^2 s + ne))$. ■

Note that s and e, which are respectively defined in Propositions 1 and 2, correspond to the same process if the oracle is a dataset. For n the number of variables, the maximum number of objects in a dataset is 2^n in the binary case. The maximum number of rules is then $k \leq 2^n$. Furthermore, each of these rules is represented by at least one swap (exactly one if $k = 2^n$). Hence, in the worst case, $s \leq 2^n$ and $e \leq 2^n$. However, in real world applications, $s < m \ll 2^n$ (with m the number of objects in a dataset) and the worst case does not hold in most of situations.

4 Experimental results

In this section, we consider that the oracle Σ is a database that contains a list of swaps. To evaluate the efficiency of our algorithm, we demonstrate its usefulness on real and on simulated datasets.

Defining an accuracy measure of a learning algorithm on such a structure is not a trivial task. A CP-net can be seen as a set of rules. A workaround to define an accuracy measure is to use the number of rules induced by the preferences of Σ which are correctly represented in $\mathcal{N}_L$ versus the total number of rules. However, even if this can be relevant in some cases, where there is just one violated rule, this measure becomes meaningless. Indeed, a violated rule cannot be represented in $\mathcal{N}_L$ and then induces a huge number of unsatisfiable swaps. In a context of learning a relevant CP-net, we rather prefer to use the number of swaps that are in agreement with $\mathcal{N}_L$ versus the total number of swaps in the whole dataset.

We use two different runs of our algorithm in order to smooth our results: the first one consists of the random generation phase of the target CP-nets or the datasets, and the second one consists of the learning phase of $\mathcal{N}_L$. We note these two runs by $k \times l$ with k the random generation phase and l the learning phase. In all the graphics in the rest of the paper, each point corresponds to a simple averaged value according to the number of runs. The standard deviation is reported as an error bar on these graphics.

We use two distinct datasets to conduct our experiments:

1. the TripAdvisor dataset [24, 25] rescaled to obtain binary attributes (considered here as variables),
2. randomly generated dataset in order to test the scalability of our algorithm, and its robustness to incoherences.

The TripAdisor[6] dataset contains about $240,000$ hotel reviews. A hotel is represented by seven rates (between 1 and 5) plus one general rate. We use this general rate as our preference relation between the reviews. To be able to learn a binary CP-net, the rates are rescaled: `1` if the rate is strictly greater than two, and `0` otherwise. We have, after this procedure, 126 different hotel reviews that induce a target CP-net $\mathcal{N}_T$ which potentially contains incoherences.

We also generate a random artificial dataset as follows: two random boolean vectors such that they form a swap (only one bit changes between both vectors) are generated along with a random score for each of these vectors. This score corresponds to our preference relation. In order to test Algorithm 1, we generate three sizes of random datasets with 50 objects (7 attributes), 500 objects (10 attributes), and 10000 objects (15 attributes). We set the size of vectors of each dataset by applying $n = \lfloor \log_2 m \rfloor + 1$, with m the number of objects in the dataset. We suppose that such a generated synthetic dataset reflects reality in the sense that a real dataset cannot contain all the possible combinations of the attribute values and some objects may not exist, i.e. $m \leq 2^{n-1} < 2^n$. A preprocessing procedure transforms a set of objects into a set of swaps.

Accuracy(%)	Agreement	Disagreement
	$p = 1$	
real hotels (126)	0.73	0.27
random hotels (50)	0.72	0.28
random hotels (500)	0.57	0.43
random hotels (10000)	0.52	0.48
	$p = 5$	
real hotels (126)	0.82	0.18
random hotels (50)	0.82	0.18
random hotels (500)	0.59	0.41
random hotels (10000)	0.51	0.49
	$p = \infty$	
real hotels (126)	0.83	0.17
random hotels (50)	0.79	0.21
random hotels (500)	0.69	0.31
random hotels (10000)	0.66	0.34
	relaxing acyclic condition	
real hotels (126)	1.00	0.00
random hotels (50)	1.00	0.00
random hotels (500)	1.00	0.00
random hotels (10000)	*too long to compute*	

Table 1. Accuracy of Algorithm 1. Results are averaged on 10×10 runs.

We firstly test the importance of having coherent preferences for the learning phase. Except for the 50 objects dataset (probably due to the few number of objects), a gap is observed between the real hotel dataset and the 500 objects dataset in Figure 3. Of course, the agreement grows with regards to the maximum number of parents. The last test consists in relaxing the acyclic condition in order to observe the its influence. In this case, we are able to exactly fit $\mathcal{N}_T$, but at the price of heavy computations.

Figure 3 depicts the results of Table 1. The accuracy increases linearly w.r.t the number of parents. Once again, one can observe a gap between the accuracy of separable ($p = 0$) and tree-shaped CP-nets ($p = 1$) for all datasets, and a decreasing accuracy between $p = 1$ and $p = 2$. The accuracy continues growing when $p > 2$. Furthermore, one can note that the learning procedure is not stable

[6] `http://times.cs.uiuc.edu/~wang296/Data/`.

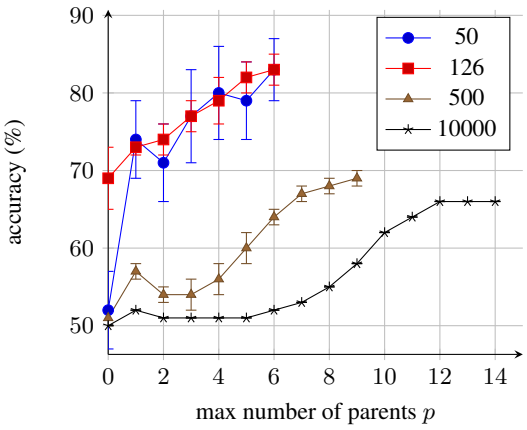


Figure 3. Learning accuracy according to the number of parent p per variable. Datasets are randomly generated except for the real 126 hotels file. Results are averaged on 5×10 runs and error bars correspond to the standard deviation of the observed values.

for small numbers of objects ($n < 500$) because of the number of variables which does not allow the algorithm to correct its mistakes.

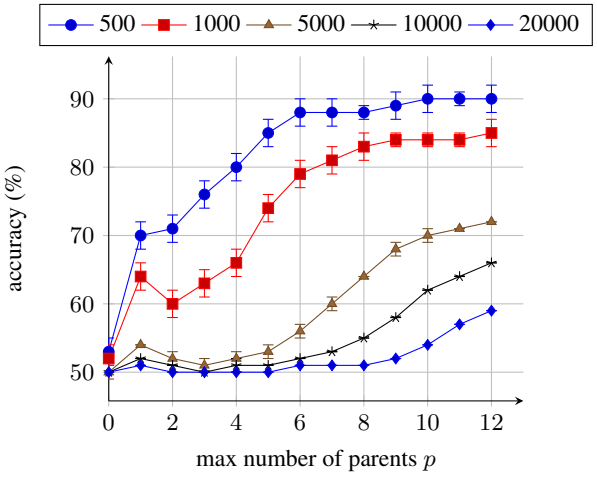


Figure 4. Learning accuracy when the number of variables is fixed ($n = 15$). We increase the number of objects in the random datasets. Results are averaged on 2×5 runs and error bars correspond to the standard deviation of the observed values.

Figure 4 shows the influence of the number of objects m on the accuracy for a fixed number of variables n. When $m \ll 2^n$, we can easily fit the structure with an accuracy greater than 80% (for the 500 and 1000 datasets). When the dataset contains a number of objects m close to the limited size 2^n, the accuracy increases until about 60%. We can observe the same phenomenon as in Figure 3, with a negative gap between $p = 1$ and $p = 2$. It occurs for all random datasets.

At last, we focus on the time taken by Algorithm 1 to learn one CP-net from a dataset. We can see in Figure 5 that this learning task is immediate for $m \leq 1000$. However, whereas we need about 20 seconds to learn a CP-net from $m = 5000$ and $p = 14$, we need more than 150 seconds to learn a CP-net from $m = 10000$ and $p = 14$. Thus, for two times more objects, we need ten times longer to com-

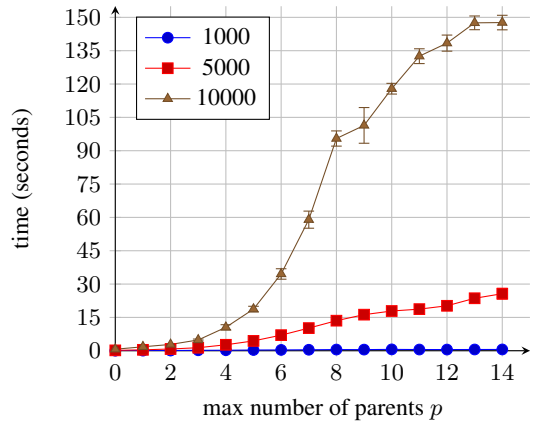


Figure 5. Learning time when the number of variables is fixed ($n = 15$). We increase the number of objects in the datasets. Results correspond to one learning execution averaged on 2×5 runs and error bars correspond to the standard deviation of the observed values.

pute it. However, the computation time increases linearly according to the number of parents: as we need to browse the whole database, the number of objects m (thus the number of swaps s that induce the rules) is a critical factor.

5 Conclusion

We presented in this paper a new algorithm for learning acyclic CP-nets, and designed a bunch of experiments to evaluate its performances on synthetic and on real datasets. They showed that despite its exponential complexity depending on the number of objects and parents (Proposition 2), our algorithm can learn in a few seconds a CP-net on random CP-nets, random datasets, or real dataset.

The main ingredient of our approach is incoherence handling in the preferences. The robustness of our algorithm to these incoherences has been evidenced by the designed experiments. For instance, on a task for hotel rating using a TripAdvisor dataset affected by noise, our algorithm achieves good accuracy results.

Future work will concern the improvement of our algorithm from different standpoints, in particular to reduce its time complexity.

A first effort will concern the implementation of a parallel version of our `searchParent` subroutine in order to decrease the learning time according to Proposition 1. This will allow to gain at least a n factor. This is especially important as recommender systems are applied in environments with massive datasets such as social networks.

A second effort will concern the improvement of several critical parts of our algorithm. The first one concerns the exhaustive nature of equivalence queries. Since we want a perfect fitting between $\mathcal{N}_T$ and $\mathcal{N}_L$ (when working with datasets) we must look over all the input data for an answer, which is very time consuming. This issue can be overcome by designing an approximate strategy, potentially with accuracy guarantees. The second critical part concerns the random nature of the counterexample returned by the equivalence queries. It may occur when the counterexample does not correspond to the real true preference rule. This leads to erroneous rules generation.

REFERENCES

[1] Eisa Alanazi, Malek Mouhoub, and Sandra Zilles, 'The complexity of learning acyclic cp-nets', in *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence, IJCAI 2016, New York, NY, USA, 9-15 July 2016*, pp. 1361–1367, (2016).

[2] Dana Angluin, 'Queries and concept learning', *Machine learning*, **2**(4), 319–342, (1988).

[3] Dana Angluin, Mārtiņš Kriķis, Robert H Sloan, and György Turán, 'Malicious omissions and errors in answers to membership queries', *Machine Learning*, **28**(2-3), 211–255, (1997).

[4] Dana Angluin and Donna K Slonim, 'Randomly fallible teachers: Learning monotone dnf with an incomplete membership oracle', *Machine Learning*, **14**(1), 7–26, (1994).

[5] Craig Boutilier, Ronen I Brafman, Carmel Domshlak, Holger H Hoos, and David Poole, 'Cp-nets: A tool for representing and reasoning with conditional ceteris paribus preference statements', *J. Artif. Intell. Res.(JAIR)*, **21**, 135–191, (2004).

[6] Ronald J Brachman, Hector J Levesque, and Raymond Reiter, *Knowledge representation*, MIT press, 1992.

[7] Yann Chevaleyre, Frédéric Koriche, Jérôme Lang, Jérôme Mengin, and Bruno Zanuttini, 'Learning ordinal preferences on multiattribute domains: The case of cp-nets', in *Preference learning*, 273–296, Springer, (2011).

[8] Yannis Dimopoulos, Loizos Michael, and Fani Athienitou, 'Ceteris paribus preference elicitation with predictive guarantees.', in *IJCAI*, volume 9, pp. 1–6. Citeseer, (2009).

[9] Alan Eckhardt and Peter Vojtás, 'How to learn fuzzy user preferences with variable objectives.', in *IFSA/EUSFLAT Conf.*, pp. 938–943, (2009).

[10] Alan Eckhardt and Peter Vojtáš, 'Learning user preferences for 2cp-regression for a recommender system', in *SOFSEM 2010: Theory and Practice of Computer Science*, 346–357, Springer, (2010).

[11] Peter C Fishburn, *Decision and value theory*, number 10, Wiley New York, 1964.

[12] Johannes Fürnkranz and Eyke Hüllermeier, *Preference learning*, Springer, 2011.

[13] Joshua T Guerin, Thomas E Allen, and Judy Goldsmith, 'Learning cp-net preferences online from user queries', in *Algorithmic Decision Theory*, 208–220, Springer, (2013).

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

[15] Frédéric Koriche and Bruno Zanuttini, 'Learning conditional preference networks', *Artificial Intelligence*, **174**(11), 685–703, (2010).

[16] Jérôme Lang and Jérôme Mengin, 'Learning preference relations over combinatorial domains', *Proceedings of NMR08*, 207–214, (2008).

[17] Juntao Liu, Chenhong Sui, Dewei Deng, Junwei Wang, Bin Feng, Wenyu Liu, and Caihua Wu, 'Representing conditional preference by boosted regression trees for recommendation', *Information Sciences*, **327**, 1–20, (2016).

[18] Juntao Liu, Yi Xiong, Caihua Wu, Zhijun Yao, and Wenyu Liu, 'Learning conditional preference networks from inconsistent examples', *Knowledge and Data Engineering, IEEE Transactions on*, **26**(2), 376–390, (2014).

[19] Juntao Liu, Zhijun Yao, Yi Xiong, Wenyu Liu, and Caihua Wu, 'Learning conditional preference network from noisy samples using hypothesis testing', *Knowledge-Based Systems*, **40**, 7–16, (2013).

[20] Ryszard S Michalski, Jaime G Carbonell, and Tom M Mitchell, *Machine learning: An artificial intelligence approach*, Springer Science & Business Media, 2013.

[21] Tuomas Sandholm, 'Expressive commerce and its application to sourcing: How we conducted $35 billion of generalized combinatorial auctions', *AI Magazine*, **28**(3), 45, (2007).

[22] Alexis Tsoukiàs, 'From decision theory to decision aiding methodology', *European Journal of Operational Research*, **187**(1), 138–161, (2008).

[23] Toby Walsh, 'Representing and reasoning with preferences', *AI Magazine*, **28**(4), 59, (2007).

[24] Hongning Wang, Yue Lu, and Chengxiang Zhai, 'Latent aspect rating analysis on review text data: a rating regression approach', in *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 783–792. ACM, (2010).

[25] Hongning Wang, Yue Lu, and ChengXiang Zhai, 'Latent aspect rating analysis without aspect keyword supervision', in *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 618–626. ACM, (2011).

A regret-based preference elicitation approach for sorting with multicriteria reference profiles

Nawal Benabbou and **Patrice Perny** and **Paolo Viappiani** [1]

Abstract. In this paper we present an incremental elicitation method to determine the importance of the coalitions of criteria in a multicriteria sorting method. The method is designed to assign alternatives to predefined categories by comparing their performance vector to reference profiles. These comparisons lead to binary preference indices that are aggregated to determine the membership of the alternatives to predefined categories. We present an active learning process to determine the weighting coefficients modeling the importance of criteria in the aggregation process. Learning examples are generated one by one and presented to the Decision Maker to efficiently reduce the uncertainty attached to criteria weights. The process is stopped when all alternatives can be assigned to a category with the desired guarantee. We present the formal elicitation method as well as numerical tests showing its practical efficiency.

Keywords: Multicriteria sorting, capacity, Choquet integral, fuzzy preference relations, ordered categories.

1 Introduction

The evaluation of alternatives is a critical task in all decision support methods. When the alternatives must be evaluated with respect to multiple criteria, several aggregation procedures have been proposed to assess the overall value of the alternatives, either to rank them by decreasing order of preference, or to assign them to predefined ordered categories on the basis of their intrinsic value. In this paper, we focus on the latter objective and we consider multicriteria evaluation methods assessing the intrinsic value of alternatives by comparing their performances to predefined reference levels.

In this family of methods, we can distinguish two approaches. The '*aggregate then compare*' approach consists in a two stage process that first aggregates the performances of every alternative to produce an overall rating (e.g. using a weighted sum) and then compares the resulting rating to reference levels to assign the alternative in a predefined category (e.g., good, medium, bad). For example, alternatives are labelled as 'good' when they rate beyond 10, as 'bad' when they rate below 5, and as 'medium' otherwise. The main drawback of this first approach is that it is often difficult to derive a significant rating from criterion values expressed on different scales.

The second approach overcomes this problem by swapping the aggregation and comparison steps; for this reason, it is named the '*compare then aggregate*' approach. This approach requires an ordered sequence of reference profiles to be defined, representing different levels of requirements in the space of criteria; these profiles are totally ordered using Pareto dominance. A category is then implicitly defined by all the alternatives that beat a given reference profile but no other reference profile ranked higher in the dominance order. Thus, reference profiles act as upper and lower bounds of categories. This way of assigning alternatives to ordered categories while using multicriteria evaluations has been initally introduced by Roy in the Electre TRI method [11, 15, 16, 4]. Then multiple variants of this method have been proposed, based on a similar scheme [12, 14, 21, 8] and known as '*multicriteria sorting methods*'. They have been widely used in various applications (see the above references) and also investigated axiomatically [2, 3].

The '*compare then aggregate*' approach requires a preference relation to be defined in the space of criteria in order to compare performance vectors of alternatives to reference profiles. This preference relation is usually defined by aggregating the preference relations derived from each criterion considered separately. The aim of this paper is to learn from examples the proper aggregation method to be used to produce the aggregated preference relation. The most common aggregation method used for aggregating preference relations derived from criteria is a weighted majority rule. In this case, the weights of criteria must be learned from examples. This problem has been studied in [12, 13, 10, 17].

More recently, an extension of this approach has been proposed in [18] based on a generalization of weighted majority using a set function (namely a *capacity*) weighting any subset of criteria. This generalization enhances the descriptive power of the weighted majority model by allowing positive or negatives synergies among criteria due to the use of a non-necessarily additive definition of criterion weights. In this paper, we generalize this approach in a number of directions. First, we consider a generalized aggregation method to assign alternatives to categories. This method, introduced in [14], uses fuzzy preference relations to compare alternatives to references profiles and defines a degree of membership of any alternative to any categories (this method will be formally specified in Section 2). Secondly, the aggregation of fuzzy preference relations is performed using a possibly non linear weighted aggregator, for example the Choquet integral that combines fuzzy preferences with a non-additive measure of the importance of criteria. The learning of classifiers based on a Choquet integral has been investigated in [9]. Our approach is different because we use here the Choquet integral to aggregate fuzzy preferences rather than criterion values. Thirdly, instead of proposing a passive learning approach aiming to assess weighting parameters from a given database of examples, we propose here an active learning of the *capacity* that progressively asks examples of assignment to the Decision Maker (DM), selected one by one to efficiently reduce the set of possible capacities until a complete assignment of the alternatives can be defined with confidence.

The paper is organized as follows: in Section 2, we recall the

[1] Sorbonne Universités, UPMC Univ Paris 06 and CNRS, LIP6, UMR 7606, Paris, France, email: name.surname@lip6.fr

main features of the assignment method introduced in [14]. Then, the incremental approach proposed to elicit the aggregation function is presented in Section 3. Finally, numerical tests showing the efficiency of the approach are presented in Section 4.

2 Sorting using reference profiles

Let K_ℓ, $\ell \in \{1,\ldots,q\}$, be a set of q ordered categories, K_1 being the best category and K_q being the worst one. Let X be the set of alternatives that must be assigned into one of these categories. Let us assume that every alternative is represented by a performance vector $x = (x_1,\ldots,x_n) \in \mathbb{R}^n$ where x_i denotes the performance of x with respect to criterion i, $i \in N = \{1,\ldots,n\}$; we assume here that criteria are to be maximized. Let us denote $\{r^0, r^1, \ldots, r^q\}$ the set of reference profiles used to defined the bounds of the categories. Each element $r^\ell \in \mathbb{R}^n$ defines the lower boundary of category K_ℓ, $\ell \in \{1,\ldots,q\}$, while $r^0 \in \mathbb{R}^n$ represents a top performance profile (not feasible) chosen to bound above all possible criterion values. These reference profiles are defined in such a way that $r_i^\ell > r_i^{\ell+1}$ for all $i \in N$ and $\ell \in \{0,\ldots,q-1\}$.

These reference profiles being defined, the principle of a preference-based assignment method is to assign x to category K_ℓ when x is preferred to r^ℓ and x is not preferred to $r^{\ell-1}$. In the method introduced in [14], this principle is implemented by comparing x to all profiles r^ℓ using a preference index defined as follows:

$$P(x, r^\ell, \theta) = f_\theta(P_1(x, r^\ell), \ldots, P_n(x, r^\ell)) \tag{1}$$

where f_θ is an aggregation function (compatible with Pareto dominance) parameterized by θ and $P_i(x, r^\ell)$ is a monocriterion preference index defined by:

$$P_i(x, r^\ell) = \begin{cases} 1 & \text{if} \quad x_i - r_i^\ell > \gamma_i^+ \\ \frac{x_i - r_i^\ell - \gamma_i^-}{\gamma_i^+ - \gamma_i^-} & \text{if} \quad \gamma_i^- < x_i - r_i^\ell \leq \gamma_i^+ \\ 0 & \text{if} \quad x_i - r_i^\ell \leq \gamma_i^- \end{cases}$$

The variation of index $P_i(x, r^\ell)$ as the difference $x_i - r_i^\ell$ increases is represented here below:

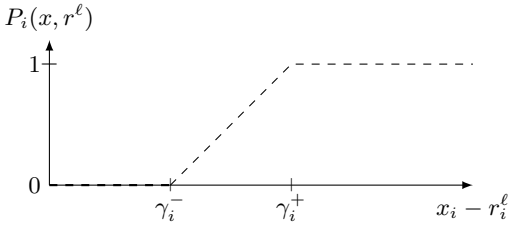


Figure 1. Preference index $P_i(x, r^\ell)$ as a function of $x_i - r_i^\ell$

This index represents the credibility of the statement "x is better than r^ℓ w.r.t criterion i". The index is maximal (i.e. equals 1) when the score difference $x_i - r_i^\ell$ exceeds the preference threshold γ_i^+. It is minimal when the difference $x_i - r_i^\ell$ falls below the indifference threshold γ_i^-. These thresholds are part of the definition of the criterion scale and are defined by the DM in such a way that $\gamma_i^+ > \gamma_i^-$. The preference threshold γ_i^+ defines the minimal difference compatible with a strict preference, whereas the indifference threshold γ_i^- represents the maximal score difference compatible with an indifference (absence of preference). Between these two thresholds, there is an area where we may hesitate between strict preference and indifference; in this area, the preference index grows linearly with $x_i - r_i^\ell$ (see Figure 1).

A standard choice for f_θ is a compromise operator, i.e. that verifies $\min_{i\in N}\{x_i\} \leq f_\theta(x) \leq \max_{i\in N}\{x_i\}$ for all $x = (x_1,\ldots,x_n) \in \mathbb{R}^n$. When using such an operator, the overall preference index $P(x, r^\ell, \theta)$ defined by Equation (1) necessary belongs to $[0, 1]$. Value 1 is achieved when the criteria are unanimously in favor of strict preference, whereas value 0 is achieved when criteria are unanimously against preference. Between these two extreme cases, the overall index measures the strength of arguments supporting the statement "x is better than r^ℓ". A common choice for f_θ is the weighted sum $f_\theta(p_1,\ldots,p_n) = \sum_{i=1}^n \theta_i p_i$ where θ_i is the weight attached to criterion i. A more interesting choice is to define f_θ as a Choquet integral [5, 6] which provides a more general and more flexible aggregator. In this case, θ is the set function defining the weight of any coalition of criteria. We will come back to this case in Section 3.2.

After the aggregation step, the index measuring on the $[0, 1]$ scale the membership of any alternative x to any category K_ℓ, $\ell \in \{1,\ldots,q\}$, is defined by:

$$m_\ell(x, \theta) = \min\Big\{P(x,r^\ell,\theta),\ 1-P(x,r^{\ell-1},\theta)\Big\} \tag{2}$$

and finally, alternative x is assigned to category K_{ℓ^*} where ℓ^* is the smallest index such that:

$$m_{\ell^*}(x, \theta) = \max_{k\in\{1,\ldots,q\}} m_k(x, \theta)$$

This procedure is the general preference-based filtering method introduced in [14]. Equation (2) ensures that, for any fixed x and θ, $m_\ell(x, \theta)$ (seen as a function of ℓ) is unimodal, with a maximum at least equal to 0.5 (for more details see [14]).

This method is typical of the '*compare then aggregate*' approach. We first compare any alternative x to all reference profiles by considering each criterion separately, which leads to indices $P_i(x, r^\ell), i \in N$. These indices are then aggregated using function f_θ to define the membership of x to any category K_ℓ. The advantage of this approach, compared to the more classical '*aggregate then compare*' approach lies in the use of reference vectors r^ℓ instead of scalar thresholds on aggregated values. This provides a finer control in the definition of category boundaries in the multiobjective space. The decision of assigning an alternative is based on the scoring vector and not on an aggregated value: two alternatives having the same "average" value but different profiles may enter into different categories, as shown in the following example:

Example 1. *Let us consider two alternatives x, y and a sequence of 3 reference profiles r^0, r^1, r^2 with the following grades on three criteria:*

	1	2	3
x	6	15	15
y	30	3	3
r^0	50	50	50
r^1	12	12	12
r^2	0	0	0

The three reference profiles allow the definition of two categories K_1 and K_2. We use here the same valuation scale with a preference threshold $\gamma_i^+ = 1$ and indifference threshold $\gamma_i^- = 0$, for all $i \in N = \{1, 2, 3\}$. Let us assume that f_θ is the weighted sum with weights $\theta = (1/3, 1/3, 1/3)$, i.e. $f_\theta(z_1, z_2, z_3) = (z_1 + z_2 + z_3)/3$ for all $z \in \mathbb{R}^n$. We obtain $P(x, r^0, \theta) = P(y, r^0, \theta) = 0$ and $P(x, r^2, \theta) =$

$P(y, r^2, \theta) = 1$. *The only difference between* x *and* y *are due to profil* r^1. *We indeed have:* $P(x, r^1, \theta) = 2/3$ *and* $P(y, r^1, \theta) = 1/3$. *Hence, using Equation (2), we get the following membership values:*

	m_1	m_2
x	$2/3$	$1/3$
y	$1/3$	$2/3$

As a consequence, x *is here assigned to category* K_1 *whereas* y *is assigned to category* K_2.

In this example, x obtains a better position than y (since K_1 is better than K_2). Note that alternatives x and y would be undiscernible with the '*aggregate then compare*' approach since they have the same average: $f_\theta(x) = f_\theta(y)$. Moreover, if the grade of y increases from 30 to 50 on criterion 1, it can easily be checked that y remains in K_2. There is no improvement despite the fact that the average score of y defined by f_θ increases. This is due to the fact that here, there is no point in improving its performance on criterion 1 since it already exceeds the value of reference profile r^1 while the weaknesses of y on criteria 2 and 3 remain. This example illustrates the non-compensatory nature of this sorting procedure, where difference of grades does not play any role beyond a given threshold; this is a clear difference with procedures based on the direct aggregation of criterion values.

3 An incremental approach for sorting alternatives using reference profiles

The procedure introduced in the previous section involves at some step an aggregation operation f_θ in which parameter θ controls the importance of criteria and coalitions of criteria in the overall assessment of alternatives. Our aim is to propose, in the framework of the approach presented above, an incremental procedure to assess parameter θ. We assume that this parameter is initially unknown and we want to use an active learning process to progressively reduce the uncertainty attached to θ. Examples will be selected one by one and presented to the DM that will be asked to classify them; these new classifications will induce constraints restricting the space of admissible θ, and the process will be repeated until being able to classify all the alternatives in X.

Our procedure relies on the notion of minimax regret, allowing to make robust decisions in face of uncertainty, and to ask informative queries to further reduce the uncertainty on the space of admissible θ. This can be seen as an adaptation to sorting problems of incremental elicitation mechanisms designed for choice problems (e.g., [20, 1]).

Whenever θ is precisely known, we wish to assign x to the category K_ℓ such that $m_\ell(x, \theta) \geq m_k(x, \theta)$ for all $k \in \{1, \ldots, q\}$. Therefore, it is natural to define the *loss* or *regret* associated to assigning x to K_ℓ rather than assigning it to K_k as:

$$\mathrm{R}(x, K_\ell, K_k, \theta) = m_k(x, \theta) - m_\ell(x, \theta).$$

When we only know that θ belong to an uncertainty set Θ, we may be interested in computing the following regrets:

Definition 1. *For any alternative* $x \in X$, *the pairwise max regret (*PMR*) of assigning* x *to the category* K_ℓ *instead of assigning it to the category* K_k *is defined by:*

$$\begin{aligned}\mathrm{PMR}(x, K_\ell, K_k, \Theta) &= \max_{\theta \in \Theta} \mathrm{R}(x, K_\ell, K_k, \theta) \\ &= \max_{\theta \in \Theta} \Big\{ m_k(x, \theta) - m_\ell(x, \theta) \Big\}.\end{aligned}$$

$\mathrm{PMR}(x, K_\ell, K_k, \Theta)$ is the maximum feasible gap between the membership indices of alternative x with respect to categories K_k and K_ℓ. It represents the worst-case loss that we may incur by assigning x to category K_ℓ instead of category K_k when parameter θ belongs to Θ.

Definition 2. *The max regret (*MR*) of assigning* $x \in X$ *to category* K_ℓ *is defined by:*

$$\mathrm{MR}(x, K_\ell, \Theta) = \max_{k \in \{1, \ldots, q\}} \mathrm{PMR}(x, K_\ell, K_k, \Theta)$$

$\mathrm{MR}(x, K_\ell, \Theta)$ is the worst-case loss that we may incur by assigning alternative x to category K_ℓ instead of any other categories. We now define the notion of *minimax regret* and the *regret-optimal category* associated to alternative $x \in X$:

Definition 3. *The minimax regret (*mMR*) of alternative* $x \in X$ *is:*

$$\mathrm{mMR}(x, \Theta) = \min_{\ell \in \{1, \ldots, q\}} \mathrm{MR}(x, K_\ell, \Theta)$$

Considering the uncertainty set Θ, a cautious decision rule would consist in assigning alternative x to the category minimizing $\mathrm{MR}(x, K_\ell, \Theta)$, named the regret-optimal category of x here below. In order to assess the maximal error on the complete assignment when using this rule, we introduce now the notion of *maximum minimax regret*:

Definition 4. *The maximum minimax regret (*MmMR*) is:*

$$\mathrm{MmMR}(X, \Theta) = \max_{x \in X} \mathrm{mMR}(x, \Theta)$$

In sorting problems, MmMR plays the role of measuring the current decision quality; in particular, if $\mathrm{MmMR} = 0$, then we know that the complete assignment is valid. However, it might be the case that the aggregate value MmMR is too large according to the DM. In that case, it is natural to consider incremental elicitation procedures to make this value decrease until a given tolerance threshold $\delta \geq 0$. This is actually possible due to the fact that the inequality $\mathrm{MmMR}(X, \Theta') \leq \mathrm{MmMR}(X, \Theta)$ is true for all $\Theta' \subseteq \Theta$, meaning that the value MmMR cannot increase when including new preference information obtained from the DM; actually, in the subsection devoted to numerical tests, we will see that, in practice, it strictly decreases when queries are chosen in a reasoned way.

As in a choice problems, comparison queries between alternatives might be asked; however, in our context, it is less straightforward to identify a pair of alternatives forming an informative preference query (i.e. which is likely to induce a regret reduction). Instead, we can ask the DM to choose, for a given alternative, the category that fits best (either among a pair of categories, among a given subset of categories, or among all possible categories). For this type of queries, we will denote $K_\ell \succsim_x K_k$ the preference for category K_ℓ over category K_k concerning the assignement of alternative $x \in X$. For choosing which query to ask next, we propose to focus on an alternative x that is associated with the largest value mMR in the current regret-optimal assignment; by asking a query involving such an alternative, we are indeed likely to reduce the value MmMR since $\mathrm{MmMR}(X, \Theta) = \mathrm{mMR}(x, \Theta)$ holds. There are at least the two following possibilities:

- we can ask the DM to assign x to the most relevant category or
- we can ask the DM to compare the regret-optimal category K_{ℓ^*} of alternative x with its *regret-maximizing adversarial category* (the one that maximizes $\mathrm{PMR}(x, K_{\ell^*}, K_k, \Theta_{\mathcal{P}})$) and state which one is the most relevant among the two.

The latter approach has the advantage of requiring less effort from the DM while focusing on the pair of categories inducing the current value MmMR.

3.1 Determination of the optimal assignment using mixed integer linear programming

Let $\mathcal{P}$ be the set gathering all preference statements of type $K_\ell \succsim_x K_k$ collected so far, and let $\Theta_\mathcal{P}$ be the set containing all parameters θ consistent with information $\mathcal{P}$, i.e. such that $m_\ell(x,\theta) \geq m_k(x,\theta)$ for all examples $K_\ell \succsim_x K_k \in \mathcal{P}$. We assume here that operator f_θ is linear in θ (e.g., a weighted sum or a Choquet integral). In order to determine the regret-optimal assignment, we need a method that efficiently computes $\mathrm{PMR}(x, K_\ell, K_k, \Theta_\mathcal{P})$ for any $x \in X$ and any $\ell, k \in \{1,\ldots,q\}$. This computation is challenging because it requires a maximization of the difference of two minima. Nevertheless, we will show that this optimization problem can be decomposed in two mixed integer linear programs. In order to do that, we need first to show that the answers to preference queries induce linear constraints over the set of parameters $\Theta_\mathcal{P}$.

Proposition 1. *Let $x \in X$ be any alternative and K_ℓ, K_k two categories such that $\ell \neq k$. If $\ell > k$, then $\Theta_{\mathcal{P}\cup K_\ell \succsim_x K_k}$ is:*

$$\Big\{\theta \in \Theta_\mathcal{P}, 1-P(x,r^{\ell-1},\theta) \geq \min\{P(x,r^k,\theta), 1-P(x,r^{k-1},\theta)\}\Big\}$$

otherwise $\Theta_{\mathcal{P}\cup K_\ell \succsim_x K_k}$ is:

$$\Big\{\theta \in \Theta_\mathcal{P}, P(x,r^{\ell},\theta) \geq \min\{P(x,r^k,\theta), 1-P(x,r^{k-1},\theta)\}\Big\}$$

Proof : If we observe that category K_ℓ is preferred to category K_k for the assignment of alternative $x \in X$, then we want to restrict $\Theta_\mathcal{P}$ to all parameters θ such that $m_\ell(x,\theta) \geq m_k(x,\theta)$, which can be rewritten $\min\{P(x,r^\ell,\theta), 1-P(x,r^{\ell-1},\theta)\} \geq \min\{P(x,r^k,\theta), 1-P(x,r^{k-1},\theta)\}$. This is actually equivalent to imposing the two following constraints:

$$P(x,r^\ell,\theta) \geq \min\{P(x,r^k,\theta),\ 1-P(x,r^{k-1},\theta)\} \quad (3)$$

$$1-P(x,r^{\ell-1},\theta) \geq \min\{P(x,r^k,\theta),\ 1-P(x,r^{k-1},\theta)\} \quad (4)$$

First, assume that K_ℓ, K_k are such that $\ell > k$. In that case, we know that we have $P_i(x,r^\ell) \geq P_i(x,r^k)$ for all $i \in N$. Therefore, since f_θ is compatible with Pareto dominance, then we have $P(x,r^\ell,\theta) \geq P(x,r^k,\theta)$ for all $\theta \in \Theta_\mathcal{P}$. As a consequence, Equation (3) holds for all $\theta \in \Theta_\mathcal{P}$ and so the associated constraint is not needed for updating the set of feasible parameters $\Theta_\mathcal{P}$ according to the observed preference $(x, K_\ell \precsim K_k)$. Hence, only Equation (4) remains in that case. Now, assuming that $\ell < k$, we have $P_i(x,r^{\ell-1}) \leq P_i(x,r^{k-1})$ for all $i \in N$. Therefore, since f_θ is compatible with Pareto dominance, then $P(x,r^{\ell-1},\theta) \leq P(x,r^{k-1},\theta)$ for all $\theta \in \Theta_\mathcal{P}$, i.e. $1-P(x,r^{\ell-1},\theta) \geq 1-P(x,r^{k-1},\theta)$. Therefore, Equation (4) holds for all $\theta \in \Theta_\mathcal{P}$ and so the associated constraint is not needed for updating the set of feasible parameters $\Theta_\mathcal{P}$ according to $K_\ell \succsim_x K_k$. Hence, only Equation (3) remains. ■

Thus, if the DM states that, for a given alternative $x \in X$, category K_ℓ is better suited than category K_k, then it is sufficient to impose the following constraint over the set of feasible parameters:

- $1-P(x,r^{\ell-1},\theta) \geq \min\{P(x,r^k,\theta), 1-P(x,r^{k-1},\theta)\}$ if $\ell > k$,
- $P(x,r^\ell,\theta) \geq \min\{P(x,r^k,\theta), 1-P(x,r^{k-1},\theta)\}$ otherwise.

These constraints can be linearized using standard linearization of the $\min$ aggregator. In particular, this is done in the following way:

If $\ell > k$, we impose:

$$\begin{cases} Mb + 1 - P(x,r^{\ell-1},\theta) \geq P(x,r^k,\theta) \\ M(1-b) + 1 - P(x,r^{\ell-1},\theta) \geq 1 - P(x,r^{k-1},\theta) \end{cases}$$

If $\ell < k$, we impose:

$$\begin{cases} Mb + P(x,r^{\ell},\theta) \geq P(x,r^k,\theta) \\ M(1-b) + P(x,r^{\ell},\theta) \geq 1 - P(x,r^{k-1},\theta) \end{cases}$$

where b is a boolean variable and M is a numerical scalar value greater than one.

Hence, we proved that $\Theta_\mathcal{P}$ can be described with linear constraints when $\mathcal{P}$ is composed of preferences of type (x, K_ℓ, K_k). Now, the following proposition proves that PMR-optimizations can be performed using mixed integer linear programming.

Proposition 2. *For any $x \in X$ and any $\ell, k \in \{1,\ldots,q\}$, we have:*

$$\mathrm{PMR}(x, K_\ell, K_k, \Theta_\mathcal{P}) = \max\{\beta_1, \beta_2\}$$

where β_1 and β_2 are respectively the optimum values of the following mixed integer linear programs:

$$\max_{\substack{\theta\in\Theta_\mathcal{P}\\ t\in\mathbb{R}}} \{t - P(x,r^\ell,\theta)\} \quad \text{s.t.} \quad t \leq P(x,r^k,\theta),\ t \leq 1-P(x,r^{k-1},\theta)$$

$$\max_{\substack{\theta\in\Theta_\mathcal{P}\\ t\in\mathbb{R}}} \{t + P(x,r^{\ell-1},\theta) - 1\} \quad \text{s.t.} \quad t \leq P(x,r^k,\theta),\ t \leq 1-P(x,r^{k-1},\theta)$$

Proof : For any alternative $x \in X$ and any two categories K_ℓ, K_k:

$$\begin{aligned} \mathrm{PMR}(x, K_\ell, K_k, \Theta_\mathcal{P}) &= \max_{\theta\in\Theta_\mathcal{P}} \mathrm{R}(x, K_\ell, K_k, \theta) \\ &= \max_{\theta\in\Theta_\mathcal{P}} \Big\{m_k(x,\theta) - m_\ell(x,\theta)\Big\} \end{aligned}$$

Since $m_k(x,\theta) = \min\{P(x,r^k,\theta), 1-P(x,r^{k-1},\theta)\}$, we can compute $\mathrm{PMR}(x, K_\ell, K_k, \Theta_\mathcal{P})$ by solving the following program:

$$\begin{aligned} \max_{\substack{\theta\in\Theta_\mathcal{P}\\ t\in\mathbb{R}}} &\Big\{t - m_\ell(x,\theta)\Big\} \\ \text{s.t. } & t \leq P(x,r^k,\theta) \\ & t \leq 1 - P(x,r^{k-1},\theta) \end{aligned}$$

This program is obtained by using standard linearization of the $\min$ aggregator. Then, we have:

$$\begin{aligned} t - m_\ell(x,\theta) &= t - \min\{P(x,r^\ell,\theta),\ 1-P(x,r^{\ell-1},\theta)\} \\ &= t + \max\{-P(x,r^\ell,\theta),\ P(x,r^{\ell-1},\theta) - 1\} \\ &= \max\{t - P(x,r^\ell,\theta),\ t + P(x,r^{\ell-1},\theta) - 1\} \end{aligned}$$

Therefore, $\mathrm{PMR}(x, K_\ell, K_k, \Theta_\mathcal{P})$ can be computed by solving the following optimization problem:

$$\begin{aligned} \max_{\substack{\theta\in\Theta_\mathcal{P}\\ t\in\mathbb{R}}} &\Big\{\max\{t - P(x,r^\ell,\theta),\ t + P(x,r^{\ell-1},\theta) - 1\}\Big\} \\ \text{s.t. } & t \leq P(x,r^k,\theta) \\ & t \leq 1 - P(x,r^{k-1},\theta) \end{aligned}$$

The result is finally obtained by interchanging the $\max$ operators. ■

Therefore, computing $\mathrm{PMR}(x, K_\ell, K_k, \Theta_\mathcal{P})$ can be easily performed by solving two mixed integer linear programs and then selecting the greatest optima.

3.2 Application to Choquet integrals

In this subsection, we will focus on a particular instance obtained by interpreting f_θ as a Choquet integral in Equation (1). This allows positive or negative synergies among arguments when aggregating preference indices $P_i(x, r^\ell)$ into an overall index $P(x, r^\ell, \theta)$. We now recall the definition of Choquet capacities and (discrete) Choquet integrals[2]. A normalized Choquet capacity v is a real-valued set-function defined on 2^N such that $v(\emptyset) = 0$, $v(N) = 1$ and $v(A) \leq v(B)$ for all $A \subseteq B \subseteq N$; value $v(A)$ is the weight attached to coalition A, for any $A \subseteq N$. The Choquet integral is then defined by:

$$C_v(x) = \sum_{i=1}^{n} \left[x_{(i)} - x_{(i-1)}\right] v(X_{(i)}) \text{ with } x_{(0)} = 0$$

where $(.)$ is a permutation of $\{1, \ldots, n\}$ which sorts the components of x by increasing order (i.e. $x_{(i)} \leq x_{(i+1)}$ for $i \in \{1, \ldots, n-1\}$) and $X_{(i)} = \{(i), \ldots, (n)\}$. In the following, the uncertain capacity function v takes the role of θ and the set of feasible parameters $\Theta_\mathcal{P}$ is the set of all normalized capacities compatible with $\mathcal{P}$. Alternative $x \in X$ is now compared to any profile r^ℓ using the *preference index*:

$$P(x, r^\ell, v) = C_v(P_1(x, r^\ell), \ldots, P_n(x, r^\ell))$$

The membership of solution x to category K_ℓ is now defined by:

$$m_\ell(x, v) = \min\{P(x, r^\ell, v),\ 1 - P(x, r^{\ell-1}, v)\}$$

In the particular case where $\Theta_\mathcal{P}$ is a set of strictly monotonic capacities (i.e. $v(A) < v(B)$ for all $A \subset B \subseteq N$), we can linearize the constraints induced by $\mathcal{P}$ in a simpler way, so that we can avoid the use of boolean variables.

Proposition 3. *For any $x \in X$ and any $\ell, k \in \{1, \ldots, q\}$:*
If $\ell > k$ and $P_i(x, r^{\ell-1}) \neq P_i(x, r^{k-1})$ for some $i \in N$, then
$\Theta_{\mathcal{P} \cup K_\ell \succsim_x K_k} = \{v \in \Theta_\mathcal{P},\ 1 - P(x, r^{\ell-1}, v) \geq P(x, r^k, v)\}$
If $\ell < k$ and $P_i(x, r^\ell) \neq P_i(x, r^k)$ for some $i \in N$, then
$\Theta_{\mathcal{P} \cup K_\ell \succsim_x K_k} = \{v \in \Theta_\mathcal{P},\ P(x, r^\ell, v) \geq 1 - P(x, r^{k-1}, v)\}$
Otherwise, $\Theta_{\mathcal{P} \cup K_\ell \succsim_x K_k} = \Theta_\mathcal{P}$.

Proof : Assume that $\ell > k$. According to Proposition 1, capacities $v \in \Theta_\mathcal{P}$ that are compatible with $K_\ell \succsim_x K_k$ are those verifying:

$$1 - P(x, r^{\ell-1}, v) \geq \min\{P(x, r^k, v), 1 - P(x, r^{k-1}, v)\} \quad (5)$$

Since $\ell > k$, we know that $P_i(x, r^{\ell-1}) \geq P_i(x, r^{k-1})$ for all $i \in N$. In the case where $P_i(x, r^{\ell-1}) = P_i(x, r^{k-1})$ for all $i \in N$, we have $P(x, r^{\ell-1}, v) = P(x, r^{k-1}, v)$, i.e. $1 - P(x, r^{\ell-1}, v) = 1 - P(x, r^{k-1}, v)$; therefore, for all $v \in \Theta_\mathcal{P}$, Equation (5) is satisfied and so $\Theta_{\mathcal{P} \cup K_\ell \succsim_x K_k} = \Theta_\mathcal{P}$. Now, consider the case where $P_i(x, r^{\ell-1}) \neq P_i(x, r^{k-1})$ for some $i \in N$. Since $v \in \Theta_\mathcal{P}$ is strictly monotonic, then C_v is strictly increasing with Pareto dominance, and so we have $P(x, r^{\ell-1}, v) > P(x, r^{k-1}, v)$, i.e. $1 - P(x, r^{\ell-1}, v) < 1 - P(x, r^{k-1}, v)$. As a consequence, Equation (5) is satisfied if and only if we have $1 - P(x, r^{\ell-1}, v) \geq P(x, r^k, v)$. Hence, $\Theta_{\mathcal{P} \cup K_\ell \succsim_x K_k} = \{v \in \Theta_\mathcal{P},\ 1 - P(x, r^{\ell-1}, v) \geq P(x, r^k, v)\}$.

Now, assume that $\ell < k$. In that case, according to Proposition 1, capacities $v \in \Theta_\mathcal{P}$ that satisfy $K_\ell \succsim_x K_k$ are those verifying:

$$P(x, r^\ell, v) \geq \min\{P(x, r^k, \theta),\ 1 - P(x, r^{k-1}, v)\} \quad (6)$$

Since $\ell < k$, we have $P_i(x, r^\ell) \leq P_i(x, r^k)$ for all $i \in N$. First, assume that $P_i(x, r^\ell) = P_i(x, r^k)$ for all $i \in N$. In that case, we have $P(x, r^\ell, v) = P(x, r^k, v)$ and so Equation (6) is verified by all capacities $v \in \Theta_\mathcal{P}$; hence $\Theta_{\mathcal{P} \cup K_\ell \succsim_x K_k} = \Theta_\mathcal{P}$. Now, assume that $P_i(x, r^\ell) \neq P_i(x, r^k)$ for some $i \in N$. Since $v \in \Theta_\mathcal{P}$ is strictly monotonic, then C_v is strictly increasing with Pareto dominance, and so we necessarily have $P(x, r^\ell, v) < P(x, r^k, v)$. Therefore, Equation (6) is satisfied if and only if $P(x, r^\ell, v) \geq 1 - P(x, r^{k-1}, v)$ and so $\Theta_{\mathcal{P} \cup K_\ell \succsim_x K_k} = \{v \in \Theta_\mathcal{P},\ P(x, r^\ell, v) \geq 1 - P(x, r^{k-1}, v)\}$. ■

Thus, according to Proposition 3, observing that category K_ℓ is preferred to category K_k for alternative x amounts to imposing:

- the linear constraint $1 - P(x, r^{\ell-1}, v) \geq P(x, r^k, v)$ if $\ell > k$ and $P_i(x, r^{\ell-1}) \neq P_i(x, r^{k-1})$ for some $i \in N$,
- the linear constraint $P(x, r^\ell, v) \geq 1 - P(x, r^{k-1}, v)$ if $\ell < k$ and $P_i(x, r^\ell) \neq P_i(x, r^k)$ for some $i \in N$ and
- no additional constraints otherwise.

As a consequence, when assuming that the DM's preferences can be modeled by a Choquet integral with a strictly monotonic capacity, the PMR-optimization consists in solving two linear programs (instead of mixed integer linear programs in the general case) and then selecting the maximum between the two optima (see Proposition 2).

4 Numerical tests

In this section, we consider datasets of alternatives uniformly drawn within $[0, 1]^n$ and simulated DMs answer to preference queries according to a randomly generated Choquet integral (defined with a strictly monotonic capactity). First, we want to compare the following query selection strategies in terms of MmMR reduction:

- **S0**: this strategy asks the DM to state which among all the categories suits most a randomly chosen alternative in the dataset.
- **S1**: this strategy asks to compare the regret-optimal category of x with its regret-maximizing adversarial category, where x is the alternative associated with the largest mMR (see Section 3).
- **S2**: this strategy asks which among all the categories suits most the alternative associated with the largest mMR value.

Linear optimizations are performed by the Gurobi solver called from a program written in Java. The reference profiles under consideration are constant utility profiles dividing the utility scale $[0, 1]$ into intervals of same size on every criterion. In Figure 2, we report the MmMR at each iteration step of the incremental procedures; results are obtained by averaging over 100runs.

First, we can see that the MmMR value reduces significantly faster with S1 and S2 than with S0; for instance, after at most 20 queries on average, the MmMR value is around 40 percent of the maximum regret in the dataset with S1 and S2, while still remaining above 80 percent with S0. Moreover, as expected, S2 turns out to be more informative than S1 (since the MmMR reduces more quickly). However, we also note that S1 needs just approximately 3 additional queries on average to achieve the same level of regret as S2. This empirical evidence suggests to use S1 instead of S2 since the former has lower cognitive cost than the latter while still being very effective (asking the DM to state which among all the categories suits most a given alternative indeed requires much more cognitive effort from the DM than the comparison of two categories).

The next numerical tests aim to evaluate the impact of the following parameters on computation times of MmMR calculations: q the number of categories, n the number of criteria, p the number of observed

[2] Refer, for instance, to [5, 6] for a much more detailed description.

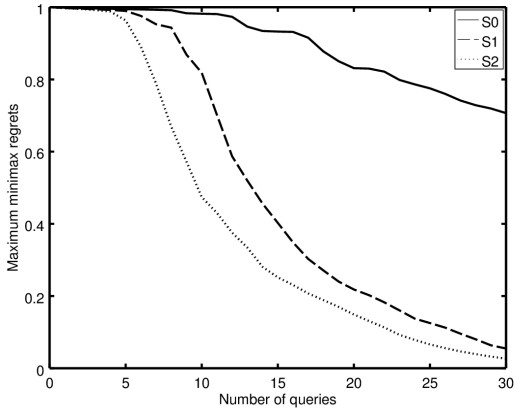


Figure 2. Maximum minimax regret reduction for strategies S0, S1 and S2 ($n = 5$, 150 alternatives, 5 categories).

preferences and the number of alternatives. In Table 1, computation times are obtained by averaging over 100 runs.

In particular, computation times drastically increase with q, the number of categories (due to the quadratic number of PMR-computations). Moreover, computation times are significantly impacted by the number of criteria. This is due to the fact that the number of variables and constraints of the linear programs grows exponentially with the number of criteria (in order to ensure monotonicity of the Choquet capacity); note however that computation times can be further reduced when considering some particular subclasses of capacities (2-additive capacities, belief functions).

Table 1. Computation times (in seconds) of MmMR calculations; results for instances with $|X| = 50, 100, 200$ are respectively given in lines 1,2,3.

$q=5$				$q=10$			
$n=5$		$n=7$		$n=5$		$n=7$	
$p=0$	$p=5$	$p=0$	$p=5$	$p=0$	$p=5$	$p=0$	$p=5$
0.8	1.4	1.8	2.3	3.5	5.6	6.7	8.6
1.1	1.8	2.1	3.3	4.6	6.8	9.1	10.8
2.2	3.5	6.9	7.1	7.6	10.7	18.2	21.1

5 Conclusion

The main advantage of our approach is that assignment queries are selected using the minimax regret criterion. The constraints derived from these examples indeed allow an efficient reduction of uncertainty where it is decisive, thus facilitating the assignment of remaining alternatives to a category. This applies to a wide family of weighted aggregation functions, including weighted sums, ordered weigthed averaging operators and Choquet integrals.

For Choquet integrals, the proposed approach is practically feasible provided that the number of criteria is not too large (about 10). For problems involving a larger number of criteria, the linear programs to be solved for computing regrets is computationally demanding, due to monotonicity constraints. In this case, a first solution consists in using capacity admitting compact representations, e.g. k-additive capacities [7] for a bounded k; in this particular case, the elicitation of the capacity remains tractable (see e.g. [19] for $k = 2$). Another interesting option would be to use fictitious alternatives with very simple profiles as learning example. This allows to drastically simplify the management of the monotonicity constraint, even with large numbers of criteria, as shown for choice problems in [1].

REFERENCES

[1] Nawal Benabbou, Patrice Perny, and Paolo Viappiani, 'Incremental Elicitation of Choquet Capacities for Multicriteria Decision Making', in *Proceedings of ECAI'14*, pp. 87–92, (2014).

[2] D. Bouyssou and Th. Marchant, 'An axiomatic approach to noncompensatory sorting methods in MCDM, I: The case of two categories', *European Journal of Operational Research*, **178**(1), 217 – 245, (2007).

[3] D. Bouyssou and Th. Marchant, 'An axiomatic approach to noncompensatory sorting methods in MCDM, II: More than two categories', *European Journal of Operational Research*, **178**(1), 246 – 276, (2007).

[4] J. Almeida Dias, J. Rui Figueira, and B. Roy, 'Electre tri-c : A multiple criteria sorting method based on characteristic reference actions', *European Journal of Operational Research*, **204**(3), (2010).

[5] M. Grabisch, 'The application of fuzzy integrals in multicriteria decision making', *European Journal of Operational Research*, **89**(3), 445–456, (1996).

[6] M. Grabisch, J.-L. Marichal, R. Mesiar, and E. Pap, *Aggregation Functions (Encyclopedia of Mathematics and Its Applications)*, Cambridge University Press, New York, NY, USA, 2009.

[7] Michel Grabisch, 'The application of fuzzy integrals in multicriteria decision making', *European journal of operational research*, **89**(3), 445–456, (1996).

[8] S. Greco, V. Mousseau, and R. Slowinski, 'Multiple criteria sorting with a set of additive value functions', *European Journal of Operational Research*, **207**(3), 1455–1470, (2010).

[9] E. Hüllermeier and A. Fallah Tehrani, 'Efficient learning of classifiers based on the 2-additive Choquet integral', in *Computational Intelligence in Intelligent Data Analysis Studies in Computational Intelligence Volume*, volume 445, pp. 17–29, (2013).

[10] A. Leroy, V. Mousseau, and M. Pirlot, 'Learning the paraeters of a multiple criteria sorting method based on a majority rule', in *proceedings of ADT'11*, volume 6992 of *LNCS*, pp. 219–233. Springer, (2011).

[11] J. Moscarola and B. Roy, 'Procdure automatique d'examen de dossiers fonde sur une segmentation trichotomique en prsence de critres multiples', *RAIRO*, **11**(2), 145–173, (1977).

[12] V. Mousseau and R. Slowinski, 'Inferring an ELECTRE TRI model from assignment examples', *Journal of Global Optimization*, **12**, 157–174, (1998).

[13] V. Mousseau, R. Slowinski, and P. Zielniewicz, 'A useroriented implementation of the ELECTRE-TRI method integrating preference elicitation support', *Computers and Operations Research*, **27**, 757–777, (2000).

[14] P. Perny, 'Multicriteria filtering methods based on concordance and nondiscordance principles', *Annals of operations Research*, **80**, 137–165, (1998).

[15] B. Roy, 'A multicriteria analysis for trichotomic segmentation problems', in *Multiple Criteria Analysis: Operational Methods*, eds., Peter Nijkamp and Jaap Spronk, pp. 245–257. Gower Press, (1981).

[16] B. Roy, 'The outranking approach and the foundations of electre methods', *Theory and Decision*, **31**, 49–73, (1991).

[17] O. Sobrie, V. Mousseau, and M. Pirlot, 'Learning a majority rule model from large sets of assignment examples', in *Proceedings of ADT'13*, pp. 336–350, (2013).

[18] O. Sobrie, V. Mousseau, and M. Pirlot, 'Learning the parameters of a non compensatory sorting model', in *Proceedings of ADT'15*, pp. 153–170, (2015).

[19] Ali Fallah Tehrani, Weiwei Cheng, Krzysztof Dembczynski, and Eyke Hüllermeier, 'Learning monotone nonlinear models using the Choquet integral', *Machine Learning*, **89**(1-2), 183–211, (2012).

[20] T. Wang and C. Boutilier, 'Incremental Utility Elicitation with the Minimax Regret Decision Criterion', in *Proceedings of IJCAI'03*, pp. 309–316, (2003).

[21] C. Zopounidis and M. Doumpos, 'Multicriteria classification and sorting methods: A literature review', *European Journal of Operational Research*, **138**(2), 229–246, (2002).

Assessing creditworthiness of companies based on lexicographic preference lists

Michael Bräuning [1] and **Tobias Keller** [2]

Abstract. We assess the creditworthiness of companies applying a machine learning algorithm for inducing generalized lexicographic preference models. The underlying model class is very transparent and reflects complex rating decisions in a simple way. The empirical performance of the lexicographic ranker is compared to commonly used models based on a sample of 1,619 Standard and Poor 's (S&P) long-term issuer rating events. We show that a reasonable assessment of companies' creditworthiness can be achieved even compared to a state-of-the-art method requiring more information than our algorithm.

1 Introduction

Many business decisions require an assessment of the counter-party's capacity to meet financial commitments. In those situations, the decision makers may refer to credit ratings as an ordinal measure for creditworthiness (or credit risk). For large companies, professional rating agencies issue so-called credit ratings. But many companies are not rated and the decision makers have to assess the creditworthiness on their own. However, since a manual in-depth analysis of the counter-parties is too costly or time-consuming and requires expert knowledge, prior literature has applied statistical and computational methods which might be more efficient. Most of the estimation methods presented in the prior literature rely on the availability of several financial key figures and/or market data and may be complex.

We present a machine learning algorithm using a lexicographic preference list for pairwise comparisons of companies regarding their creditworthiness. This algorithm requires only little information that is publicly available for most companies. A lexicographic decision between two alternatives is based on the first attribute the alternatives differ on without considering any succeeding attributes. In the literature, this is called a *non-compensatory* strategy [Gigerenzer and Selten(2001), Bouyssou and Vansnick(1986)]. As a consequence, our lexicographic heuristic supports fast and frugal decision making: once a lexicographic preference structure has been learned, subsequent decisions can typically be made using only few attributes. Furthermore, the induced lists may reveal useful insights into partially intransparent credit rating methodologies.

2 Related work and method

Computational methods are often used to assess the creditworthiness of companies. Prior literature has compared statistical models and/or machine learning algorithms to predict credit ratings. Earlier studies use ordinary least squares (OLS) regressions (e.g. [Horrigan(1966)], [Pogue and Soldofsky(1969)], and [West(1970)]) or multiple discriminant analysis (MDA, e.g. [Pinches and Mingo(1973)], [Altman and Katz(1976)], and [Belkaoui(1980)]). Multinomial probit or logit regressions have been used in some studies (e.g. [Gentry et. al(1988)]) but since the late 1970s, studies have been focusing on ordered probit or ordered logit regressions (e.g. [Kaplan and Urwitz(1969)], [Ashbaugh-Skaife et al.(2006)] or [Caporale et al.(2012)]) as they have been shown to perform better than OLS and MDA ([Ederingtonl(1985)]).

Another strand of literature predicts ratings using machine learning or artificial intelligence algorithms. Quite a few studies apply neural networks algorithms (e.g. [Kwon et al.(1997)]). Other studies use inductive learning ([Shaw and Gentry(1990)]) or case-based reasoning (e.g. [Kim and Han(2001)]). Several studies make use of support vector machines (SVM) (e.g. [Lee(2007)]). A related stream of literature uses classification algorithms to predict credit scorings - see [Lessmann et al.(2015)] for a literature review. In contrast to those algorihtms, the proposed method is not based on any scores.

To the best of our knowledge, lexicographic heuristics have not yet been considered in assessing creditworthiness and might provide an interesting alternative due to their clear structure. In preference modeling, lexicographic orders have already been used since the seventies of the last century ([Fishburn(1974)]), whereas this type of structure has only recently attracted attention in the field of machine learning. Flach and Matsubara develop a lexicographic ranker and show experimentally that the ranker is competitive to decision trees and the Naive Bayes classifier in terms of ranking performance ([Flach and Matsubara(2007)]). [Kohli and Jedidi(2007)] present two variants of a lexicographic preference and introduce a greedy algorithm to derive lexicographic models including an empirical validation on a small data set. Further work on learning lexicographic orders is done by [Schmitt and Martignon(2006)], [Dombi et al.(2007)], and [Yaman et al.(2008] but based on rather simplistic assumptions. More general models have been studied by [Booth(2010)] and inspired parts of our work.

This extended abstract mainly builds on our own previous work [Bräuning and Hüllermeier(2012), Bräuning et al.(2016)] and can be seen as a promising application thereof.

We proceed from an attribute-value representation of decision alternatives. A lexicographic order is a total order defined in terms of a total order on a set of attributes, i.e., a ranking of the attributes, and a total order on each attribute domain.

The assumption of a lexicographic representation of an order relation can be seen as an inductive bias restricting the model space. In our approach, we work with generalized lexicographic

[1] Department of Mathematics and Computer Science, Philipps-Universität Marburg, Germany, email: michael_braeuning@gmx.de
[2] Department of Business Administration and Economics, Justus-Liebig-Universität Gießen, Germany, email: mail@tobiaskeller.net

orders that allow for *attribute grouping* [Wilson(2009)]: Several (one-dimensional) attributes can be grouped into a single high-dimensional attribute, and preferences can be specified on the Cartesian product of the corresponding domains. Obviously, attribute grouping significantly increases the expressivity of the model class.

Referring to the above definition of lexicographic orders combined with attribute grouping, we end up with what we call a *lexicographic preference list*, or LP list for short. Graphically, this is a structure in the form of a list, in which every item is labelled with a subset of attributes and a total order on the Cartesian product of the corresponding attribute domains.

We are interested in finding a ranking function represented in the form of a LP list, which generalizes beyond a set of instance pairs. Problems of this kind have recently been studied in the realm of preference learning [Hüllermeier et al.(2008)], where the problem of learning ranking functions from pairwise comparisons is known as *object ranking* [Cohen et al.(1998)]. The problem of *multipartite ranking* [Fürnkranz et al.(2009)] is quite similar to the one of object ranking. However, training data is not directly presented in the form of pairwise preferences which is required as input format. Instead, it is given in the form of instances together with a categorization, where the categorization is an ordinal scale equipped with an order relation (such as a company rating). Yet, training information of that kind can be transformed into data of the desired format.

Our learning algorithm LPLL, which is short for *lexicographic preference list learner*, is in detail described in [Bräuning et al.(2016)]. However, we would like to highlight that lexicographic orders are essentially restricted to attributes with finite domains, typically comprising only a small to moderate number of values. Consequently, attibutes with continuous domains need to be discretized in advance. Therefore, we adopt the discretization technique by [Fayyad and Irani(1993)] and incorporate it in the LPLL.

LPLL comprises two important parameters, namely, r_{max}, controlling the number of bins for the discretization, and g_{max}, controlling the number of attributes grouped together. Due to reasons of interpretability, and to assure a sufficiently simple model, we restrict them to $r_{max} < 5$ and $g_{max} < 4$. The optimal values of these parameters may depend on the data. We estimate these values empirically by cross-validation.[3]

Finally, LPLL induces a LP list which can be used to compare new pairs of (query) instances. More generally, the induced LP list implements a ranking function that can be used for ranking any query set. To this end, the pairwise comparison realized by the LP list can be embedded in any standard sorting algorithm. Note that, the result of the sorting procedure will in general only be a weak order.

3 Results

Based on our literature review, we select attributes in order to assess and compare the companies' creditworthiness. We use financial accounting data from the most recent annual report with respect to the rating event. Variables in scope comprise, for example, return on assets (ROA), leverage ratio or interest coverage ratio. In addition, we investigate the scenario of limited data availability where we consider only the five most commonly available variables plus the country of incorporation. We apply the same setting as discussed in our previous work ([Bräuning et al.(2016)]) and compare LPLL to the most relevant methods identified within the previous section. The ranking performance is measured in terms of the so-called Concordance Index (C-Index), which is a generalization of the AUC for the case of multi-partite data. Table 1 shows the results based on a 10-times 10-fold cross validation indicating that variable grouping improves the performance of the identified LP lists.

LPLL C-Index	r^* / g^*_{max}	SVM^{Rank}	ordered probit	ordered logit
Full set of attributes (N = 1,361)				
thereof: non-financial companies (N = 920)				
.7680 $\pm$.0382	3.52 / 1.91	**.8093** $\pm$.0031	.7865 $\pm$.0413	.7897 $\pm$.0284
thereof: financial companies (N = 441)				
.6920 $\pm$.0513	2.12 / 2.09	**.7773** $\pm$.0041	.7509 $\pm$.0297	.7490 $\pm$.0284
Top-6 attributes available (N = 1,619)				
thereof: non-financial companies (N = 1,072)				
.6743 $\pm$.0350	3.13 / 1.96	**.6918** $\pm$.0030	.6568 $\pm$.0272	.6612 $\pm$.0272
thereof: financial companies (N = 547)				
.6962 $\pm$.0370	2.02 / 1.47	**.7090** $\pm$.0059	.6423 $\pm$.0490	.6419 $\pm$.0495

Table 1. Average performance in terms of C-Index based on a 10-times 10-fold cross-validation where r^* and g^*_{max} represent the average values of the individual best binning and grouping parameters.

Not surprisingly, SVM^{Rank} and ordered probit and logit outperform LPLL. What needs to be considered, however, is the fact that the benchmark methods use the original numerical features, and therefore do not suffer from any loss of information due to discretization. Moreover, these methods consider every attribute given to score an instance whereas the average number of attributes taken into account in a LP list is far less ([Bräuning et al.(2016)]).

Referring to the more realistic scenario of limited data availablity, it turns out that LPLL outperforms ordered probit and logit. Additionally, the difference in performance is surprisingly small compared to SVM^{Rank}.

4 Conclusion

In this extended abstract we demonstrate that a reasonable assessment of creditworthiness can be achieved using a heuristic requiring only little information that is publicly available for most companies. Consequently, the relative performance of our lexicographic ranker improves in situations of restricted data availability.

Although we mainly use our lexicographic ranker for pairwise comparisons of companies with regard to their creditworthiness measured by credit ratings, future research could use the algorithm to derive a ranking for unrated companies. Since our lexicographic model requires only few attributes to arrive at a reasonable assessment of creditworthiness, it could be applied to companies for which data availability is restricted.

Finally, we are going to extend LPLL in order to enable ordinal classification and, eventually, the prediction of ratings in addition to the ranking of companies.

REFERENCES

[Altman and Katz(1976)] Altman, E., Katz, S., 1976. Statistical bond rating classification using financial and accounting data. Proceedings of the conference on topical research in accounting, pp. 205–239.

[3] If cross-validation is used to evaluate the performance of LPLL, as will be done in our case study, an extra inner loop is needed, leading to a nested cross-validation.

[Ashbaugh-Skaife et al.(2006)] Ashbaugh-Skaife, H., Collins, D., LaFond, R., 2006.The effects of corporate governance on firms' credit ratings: Conference Issue on Implications of Changing Financial Reporting Standards. Journal of Accounting and Economics 42 (1-2), pp. 203–243.

[Belkaoui(1980)] Belkaoui, A., 1980. Industrial bond ratings: a new look.Financial Management 9, pp. 44–51.

[Booth(2010)] Booth, R., Chevaleyre, Y., Lang, J., Mengin, J., Sombattheera, C., 2010. Learning conditionally lexicographic preference relations. Proceedings of the 19th European Conference on Artificial Intelligence, pp. 269–274.

[Bouyssou and Vansnick(1986)] Bouyssou, D., Vansnick, J., 1986. Noncompensatory and generalized noncompensatory preference structures. Theory and Decision 21, pp. 251–266.

[Bräuning and Hüllermeier(2012)] Bräuning, M., Hüllermeier, E., 2012. Learning conditional lexicographic preference trees. In: Proceedings of the European Conference on Artificial Intelligence - Workshop on Preference Learning: Problems and Applications in AI.

[Bräuning et al.(2016)] Bräuning, M., Hüllermeier, E., Keller, T., Glaum, M., 2016. Lexicographic preferences for predictive modeling of human decision making: A new machine learning method with an application in accounting. European Journal of Operational Research. http://dx.doi.org/10.1016/j.ejor.2016.08.055

[Caporale et al.(2012)] Caporale, G., Matousek, R., Stewart, C., 2012. Ratings assignments: Lessons from international banks. Journal of International Money and Finance 31 (6), pp. 1593–1606.

[Cohen et al.(1998)] Cohen, W., Schapire, R., Singer, Y., 1998. Learning to order things. In: Jordan, M., Kearns, M., Solla, S. (Eds.), Advances in Neural Information Processing Systems. The MIT Press.

[Dombi et al.(2007)] Dombi, J., Imreh, C., Vincze, N., 2007. Learning lexicographic orders. European Journal of Operational Research 183 (2), pp. 748–756.

[Ederingtonl(1985)] Ederington, L., 1985. Classification models and bond ratings. Financial Review 20 (4), pp. 237–262.

[Fishburn(1974)] Fishburn, P., 1974. Lexicographic Orders, Utilities and Decision Rules: A Survey. Management Science 20 (11), pp. 1442–1471.

[Flach and Matsubara(2007)] Flach, P. and Matsubara, E.i, 2007.A Simple Lexicographic Ranker and Probability Estimator. Proceedings of the 18th European conference on Machine Learning, pp. 575–582.

[Fayyad and Irani(1993)] Fayyad, U. M., Irani, K. B., 1993. Multi-interval discretization of continuous-valued attributes for classification learning. In: International Joint Conference on Artificial Intelligence. pp. 1022–1029.

[Fürnkranz et al.(2009)] Fürnkranz, J., Hüllermeier, E., Vanderlooy, S., 2009. Binary decomposition methods for multipartite ranking. In: Proceedings of the European Conference on Machine Learning and Knowledge Discovery in Databases. Bled, Slovenia.

[Gentry et. al(1988)] Gentry, J., Whitford, D., Newbold, P., 1988. Predicting Industrial Bond Ratings with a Probit Model and Fund Flow Components. Financial Review 23 (3), pp. 269–286.

[Gigerenzer and Selten(2001)] Gigerenzer, G., Selten, R., 2001. Bounded Rationality: The adaptive toolbox. Cambridge: The MIT Press.

[Horrigan(1966)] Horrigan, J., 1966. The determination of long-term credit standing with financial ratios. Journal of Accounting Research 4, pp. 44–62.

[Hüllermeier et al.(2008)] Hüllermeier, E., Fürnkranz, J., Cheng, W., Brinker, K., 2008. Label ranking by learning pairwise preferences. Artificial Intelligence 172, pp. 1897–1917.

[Kaplan and Urwitz(1969)] Kaplan, R. and Urwitz, G., 1969. Statistical Models of Bond Ratings: A Methodological Inquiry. The Journal of Business 52 (2), pp. 231–261.

[Kim and Han(2001)] Kim, K., Han, I., 2001. The cluster-indexing method for case-based reasoning using self-organizing maps and learning vector quantization for bond rating cases. Expert Systems with Applications 21 (3), pp. 147–156.

[Kohli and Jedidi(2007)] Kohli, R., Jedidi, K., 2007. Representation and inference of lexicographic preference models and their variants. Marketing Science 26 (3), pp. 380–399.

[Kwon et al.(1997)] Kwon, Y. and Han, I. and Lee, K., 1997. Ordinal Pairwise Partitioning (OPP) Approach to Neural Networks Training in Bond rating. Intelligent Systems in Accounting, Finance & Management 6 (1), pp. 23–40.

[Lee(2007)] Lee, Y., 2007. Application of support vector machines to corporate credit rating prediction. Expert Systems with Applications 33 (1), pp. 67–74.

[Lessmann et al.(2015)] Lessmann, S., Baesens, B., Seow, H. and Thomas, L., 2015. Benchmarking state-of-the-art classification algorithms for credit scoring: an update of research. European Journal of Operational Research 247 (1), pp. 124–136.

[Pinches and Mingo(1973)] Pinches, G., Mingo, K., 1973.A Multivariate Analysis of Industrial Bond Ratings. The Journal of Finance 28 (1), pp. 1–18.

[Pogue and Soldofsky(1969)] Pogue, T., Soldofsky, R., 1969. What's in a Bond Rating. The Journal of Financial and Quantitative Analysis 4 (2), pp. 201–228.

[Schmitt and Martignon(2006)] Schmitt, M. and Martignon, L., 2006. On the Complexity of Learning Lexicographic Strategies. Journal of Machine Learning Research 7, pp. 55–83.

[Shaw and Gentry(1990)] Shaw, M. and Gentry, J., 1990. Inductive learning for risk classification. IEEE Intelligent Systems 5 (1), pp. 74–53.

[West(1970)] West, R., 1970. An alternative approach to predicting corporate bond ratings. Journal of Accounting Research 8, pp. 118–125.

[Wilson(2009)] Wilson, N., 2009. Efficient Inference for Expressive Comparative Preference Languages. Proceedings of the21st international joint conference on Artificial Intelligence, pp. 961–966.

[Yaman et al.(2008] Yaman, F., Walsh, T., Littman, M., desJardins, M., 2008. Democratic approximation of lexicographic preference models. Proceedings of the 25th international conference on Machine learning, pp. 1200–1207.

Possible and necessary labels in K-nn procedures to query partially labelled data

Vu-Linh Nguyen and **Sébastien Destercke** and **Marie-Hélène Masson**[1]

Abstract. When learning from partially labelled data (i.e., given as a subset of possible labels containing the true one), an issue that naturally arise is to determine which data should be queried to improve the accuracy of predictions, or in other word to determine an order between the partial labels to be queried. An answer to that question is to query the data that would induce the highest number of ambiguous predictions. In the K-nn case, studied in this paper, this question is similar to determining possible and necessary winners in plurality voting. In this paper, we discuss this connection as well as its use in partial label querying.

1 Introduction

The problem of learning from partial labels is known under various names: "learning with partial labels" [2], "learning with ambiguous labels" [3] or "superset label learning" [5]. In these works, authors have either proposed general schemes to learn from partial labels, for instance by adapting the loss function to partial labels [2], or to adapt specific algorithms (e.g. K-nn or decision trees) to the case of partial labels [3].

In general, the less partial are the labels, the better these techniques will perform. In the spirit of active learning techniques, this work addresses the problem of finding which partial labels should be disambiguated by an oracle (expert) in order to achieve better performances. In this work, we adopt a robust view consisting in considering all possible replacements of the partial labels instead of making any posterior probability of true labels.

In order to find which instance to query, we propose in Section 2 a general scheme based on measuring the potential impact of knowing the true label of a given instance. We then propose a specific measure to assess this impact which considers whether a partial label introduces some ambiguity in the decision, using some notions issued from social choice theory [6] in Section 3.

2 General scheme

In our setting, we assume that we have one training set $\mathbf{D} = \{(\mathbf{x}_n, \mathbf{y}_n) | n = 1, \ldots, N\}$ used to make predictions, with $\mathbf{x}_n \in \mathbb{R}^P$ the features and $\mathbf{y}_n \subseteq \Omega = \{\lambda_1, \ldots, \lambda_M\}$ potentially imprecise labels. As usual when working with partial labels [2], we assume that $\mathbf{y}_n$ contains the true label. We also assume that we have one unlabelled target set $\mathbf{T} = \{\mathbf{t}_j | j = 1, \ldots, T\}$ that will be used to determine the partial labels to query and can be defined differently based on the usage purposes.

[1] UMR CNRS 7253 Heudiasyc, University of Technology of Compiegne, email:{linh.nguyen, sebastien.destercke, mylene.masson}@hds.utc.fr

When data $\mathbf{y}_n$ are precise, the decision $h(\mathbf{t})$ taken by the K-nn procedure is given by

$$h(\mathbf{t}) = \arg\max_{\lambda \in \Omega} \sum_{\mathbf{x}_k^{\mathbf{t}} \in \mathbf{N_t}} w_k^{\mathbf{t}} \mathbb{1}_{\lambda = \mathbf{y}_k^{\mathbf{t}}} \quad (1)$$

where $\mathbf{N_t} = \{\mathbf{x}_1^{\mathbf{t}}, \ldots, \mathbf{x}_K^{\mathbf{t}}\}$ and $\mathbf{w_t} = \{w_1^{\mathbf{t}}, \ldots, w_K^{\mathbf{t}}\}$ are the K nearest neighbours of $\mathbf{t}$ and their associated weights, respectively. Equation (1) directly extends to the partial label case [3] by replacing $\lambda = \mathbf{y}_k^{\mathbf{t}}$ by $\lambda \in \mathbf{y}_k^{\mathbf{t}}$, however this comes down to consider a particular replacement of partial labels.

This is useful when the goal is to determine a single prediction from partial labels, however if the aim is to identify those partial labels that would be the most useful to query, it seems preferable to identify which data in $\mathbf{D}$ makes predictions in $\mathbf{T}$ ambiguous, with the idea that the more ambiguity they induce, the more interesting it is to know their label.

In the next section, we discuss the problem of determining whether $\mathbf{D}$ induces some ambiguity on a particular instance $\mathbf{t}$, and also a way to quantify whether querying a particular instance $\mathbf{x}_n$ would have an effect on this ambiguity. We will see that this is strongly connected to issues from social choice theory [6].

3 Indecision-based querying criteria

In this section, we present an effect score that is based on whether a partial labelled instance $\mathbf{x}_n$ introduces some ambiguity in the decision about an instance $\mathbf{t}$. We will first define what we mean by ambiguity.

3.1 Ambiguous instance: definition

In the K-nn algorithm, each neighbour can be seen as a (weighted) voter in favor of his preferred class. Partial labels can then be assimilated to voters providing incomplete preferences. For this reason, we will define ambiguity by using ideas issued from majority voting procedure with incomplete preferences [4]. More precisely, we will use the notions of necessary and possible winners of such a voting procedure to determine when a decision is ambiguous.

For an instance $\mathbf{t}$ with $\mathbf{N_t} = \{\mathbf{x}_1^{\mathbf{t}}, \ldots, \mathbf{x}_K^{\mathbf{t}}\}$, we will denote by $\mathbf{L_t} = \{(l_1^{\mathbf{t}}, \ldots, l_K^{\mathbf{t}}) | l_k^{\mathbf{t}} \in \mathbf{y}_k^{\mathbf{t}}\}$ the set of possible selections of $\mathbf{N_t}$ with cardinality $|\mathbf{L_t}| = \prod_{k=1}^{K} |\mathbf{y}_k^{\mathbf{t}}|$. For a given selection $\mathbf{l}^{\mathbf{t}} \in \mathbf{L_t}$, the corresponding winner(s) of the voting procedure is (are)

$$\widehat{\lambda}_{\mathbf{l}^{\mathbf{t}}} = \arg\max_{\lambda \in \Omega} \sum_{k=1}^{K} w_k^{\mathbf{t}} \mathbb{1}_{l_k^{\mathbf{t}} = \lambda}$$

with $w_k^{\mathbf{t}}$ the weight corresponding to the kth neighbor. Let us note that the $\arg\max$ can return multiple labels (we do not break ties).

We can now define the possible ($\mathbf{PL_t}$) and necessary label sets ($\mathbf{NL_t}$) of $\mathbf{t}$ as follows:

$$\mathbf{PL_t} = \{\lambda \in \Omega | \exists \mathbf{l^t} \in \mathbf{L_t} \text{ s.t } \lambda \in \widehat{\lambda}_{\mathbf{l^t}}\} \quad (2)$$

and

$$\mathbf{NL_t} = \{\lambda \in \Omega | \forall \mathbf{l^t} \in \mathbf{L_t}, \lambda \in \widehat{\lambda}_{\mathbf{l^t}}\}, \quad (3)$$

which are nothing else but the set of possible and necessary winners in social choice theory. By definition, we have $\mathbf{NL_t} \subseteq \mathbf{PL_t}$. Given a target instance, we adopt the following definition of ambiguity.

Definition 1. *A target instance* $\mathbf{t}$ *is called ambiguous if* $\mathbf{NL_t} \neq \mathbf{PL_t}$.

The ideal situation is to have $\mathbf{PL_t} = \mathbf{NL_t}$ and $|\mathbf{PL_t}| = 1$, since in this case the decision is uniquely defined.

3.2 Ambiguous instance: computational issues

Let us first provide some definitions. For each $\lambda \in \Omega$, we define the minimum and maximum scores as

$$S^{\min}(\lambda) = \sum_{k=1}^{K} w_k^{\mathbf{t}} \mathbb{1}_{\lambda = \mathbf{y}_k^{\mathbf{t}}} \text{ and } S^{\max}(\lambda) = \sum_{k=1}^{K} w_k^{\mathbf{t}} \mathbb{1}_{\lambda \in \mathbf{y}_k^{\mathbf{t}}},$$

whose computation can be done in linear time.

Computing $\mathbf{NL_t}$ The problem of determining $\mathbf{NL_t}$ is actually very easy, as it is known [4] that

$$\mathbf{NL_t} = \left\{\lambda |\ S^{min}(\lambda) \geq S^{max}(\lambda^{'}), \forall \lambda^{'} \neq \lambda, \lambda^{'} \in \Omega\right\}$$

Computing $\mathbf{PL_t}$ Determining $\mathbf{PL_t}$ is in practice much more difficult. In the unweighted case (all weights in $\mathbf{w_t}$ equals), known results indicate [1, 7] that $\mathbf{PL_t}$ can be determined in cubic (hence polynomial) time, solving a maximum flow problem and using the fact that when votes are (made) unitary, the solution of this flow problem is integer-valued (due to the submodularity of the constraint matrix).

However, when votes are non-unitary, or when weights are different, this result does not hold anymore, and the problem appears to be NP-hard. In addition to that, in our setting we can have to evaluate $\mathbf{PL_t}$ a high number of times (in contrast with what happens in social choice, where $\mathbf{PL_t}$ and $\mathbf{NL_t}$ have to be evaluated at most a few times), hence even a cubic algorithm may have a prohibitive computational time. A computationally cheap approximation is then to consider the set

$$\mathbf{APL_t} = \left\{\lambda | S^{max}(\lambda) \geq \max_{\lambda' \in \Omega_{\mathbf{t}}} S^{min}(\lambda^{'}), \forall \lambda^{'} \neq \lambda, \lambda^{'} \in \Omega\right\}$$

3.3 Effect of a query on ambiguous instances

Once we know which predictions are ambiguous, it remains to determine which instances in $\mathbf{D}$ we should query in order to reduce the most this ambiguity. We adopt a simple scheme to do that: for an instance $\mathbf{x}_n$, we determine a local score $f_{\mathbf{x}_n}(\mathbf{t})$ determining whether querying $\mathbf{x}_n$ can reduce our ambiguity on $\mathbf{t}$, and then aggregate this local score into a global score

$$f_{\mathbf{x}_n}(\mathbf{T}) = \sum_{\mathbf{t} \in \mathbf{T}} f_{\mathbf{x}_n}(\mathbf{t}), \quad (4)$$

over the whole set $\mathbf{T}$ which is simply the sum of local scores over each instance t. Let us denote by $\mathbf{NL}_{\mathbf{t}}^{q_n}$, $\mathbf{APL}_{\mathbf{t}}^{q_n}$ and $\mathbf{PL}_{\mathbf{t}}^{q_n}$ the sets obtained if we learn $\mathbf{y}_n = \lambda$. Then we can define the local scores

$$f_{\mathbf{x}_n}^{APL}(\mathbf{t}) = \begin{cases} 1 \text{ if } \exists \lambda \text{ s.t. } \mathbf{NL}_{\mathbf{t}}^{q_n} \neq \mathbf{NL_t} \text{ or } \mathbf{APL}_{\mathbf{t}}^{q_n} \neq \mathbf{APL_t}, \\ 0 \text{ else.} \end{cases}$$

and

$$f_{\mathbf{x}_n}^{PL}(\mathbf{t}) = \begin{cases} 1 \text{ if } \exists \lambda \text{ s.t. } \mathbf{NL}_{\mathbf{t}}^{q_n} \neq \mathbf{NL_t} \text{ or } \mathbf{PL}_{\mathbf{t}}^{q_n} \neq \mathbf{PL_t}, \\ 0 \text{ else.} \end{cases}$$

with $f_{\mathbf{x}_n}^{PL}(\mathbf{t})$ being more complex to estimate than $f_{\mathbf{x}_n}^{APL}(\mathbf{t})$, for similar reasons as the one mentioned in Section 3.2. In particular, we can show that there are easy ways to estimate $f_{\mathbf{x}_n}^{APL}(\mathbf{t})$.

We can then use any of these functions to determine the global score $f_{\mathbf{x}_n}(\mathbf{T})$, and which instance $\mathbf{x}_n$ to query.

4 Ongoing works

Our current work concerns the investigation of computational issues as well as experimental comparisons of different approaches:

- we are currently comparing different querying schemes to the use of $f_{\mathbf{x}_n}^{PL}(\mathbf{t})$ and $f_{\mathbf{x}_n}^{APL}(\mathbf{t})$, such as classical active learning, random querying, querying the most partial instances first, ... in order to know whether identifying ambiguous situations, which are computationally more difficult to identify, is beneficial to the learning procedure. Current results show that, indeed, there is an advantage in identifying those instances that induce a lot of ambiguity;
- we are also currently investigating the computational problem of determining $\mathbf{PL_t}$ in the weighted case. First results indicate that the problem is NP-hard (it seems to be reducible to a 3-dimensional matching problem), yet a refined complexity analysis is necessary to identify whether it is an important issue for our case.

REFERENCES

[1] Nadja Betzler and Britta Dorn, 'Towards a dichotomy for the possible winner problem in elections based on scoring rules', *Journal of Computer and System Sciences*, **76**(8), 812–836, (2010).

[2] Timothee Cour, Ben Sapp, and Ben Taskar, 'Learning from partial labels', *The Journal of Machine Learning Research*, **12**, 1501–1536, (2011).

[3] Eyke Hüllermeier and Jürgen Beringer, 'Learning from ambiguously labeled examples', *Intelligent Data Analysis*, **10**(5), 419–439, (2006).

[4] Kathrin Konczak and Jérôme Lang, 'Voting procedures with incomplete preferences', in *Proc. IJCAI-05 Multidisciplinary Workshop on Advances in Preference Handling*, volume 20. Citeseer, (2005).

[5] Liping Liu and Thomas Dietterich, 'Learnability of the superset label learning problem', in *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*, pp. 1629–1637, (2014).

[6] Hervé Moulin, Felix Brandt, Vincent Conitzer, Ulle Endriss, Jérôme Lang, and Ariel D Procaccia, *Handbook of Computational Social Choice*, Cambridge University Press, 2016.

[7] Lirong Xia and Vincent Conitzer, 'Determining possible and necessary winners given partial orders', *Journal of Artificial Intelligence Research*, 25–67, (2011).

Preference-based Reinforcement Learning using Dyad Ranking

Dirk Schäfer[1] and **Eyke Hüllermeier**[2]

Abstract. Preference-based reinforcement learning has recently been introduced as a generalization of conventional reinforcement learning. Instead of numerical rewards, which are often difficult to specify, the former assumes weaker feedback in the form of qualitative preferences between states or trajectories. A specific realization of preference-based reinforcement learning is approximate policy iteration using label ranking. We propose an extension of this method, in which label ranking is replaced by so-called *dyad ranking*. The main advantage of this extension is the ability of dyad ranking to learn from feature descriptions of actions, which are often available in reinforcement learning. Several simulation studies are conducted to confirm the usefulness of the approach.

1 INTRODUCTION

Reinforcement learning (RL) is an established machine learning methodology for modeling and optimizing the behavior of an autonomous agent acting in a dynamic environment [13]. A key component of RL is a numerical reward function that is used to provide positive or negative (and possibly delayed) feedback signals for the agent's actions. This quest for numerical information impedes the use of RL in situations where precise rewards are difficult to specify.

This was the main motivation for *preference-based reinforcement learning* (PBRL), which has recently been introduced as a generalization of conventional RL [1, 2]. Instead of numerical rewards, it assumes weaker feedback in the form of qualitative preferences between states or trajectories. A specific realization of preference-based reinforcement learning is a combination of *approximate policy iteration* [3] and a preference-learning method called *label ranking* [15]. Roughly speaking, label ranking is used to generalize training information of the form "in state $\boldsymbol{s}$, taking action $\boldsymbol{a}$ appears to be better than action $\boldsymbol{a}'$", so that a ranking of all available actions can be predicted for all states of the agent's state space.

In this paper, we propose an extension of this method, in which label ranking is replaced by so-called *dyad ranking* [9]. The main advantage of this extension is the ability of dyad ranking to learn from feature descriptions of actions, i.e., properties of the actions $\boldsymbol{a}$, which are often available in reinforcement learning. This is not possible in standard label ranking, where choice alternatives are merely identified by their name but not characterized in terms of attributes. Our speculation is that exploiting feature-descriptions will improve learning by generalizing, not only over the state space, but also over the action space.

The paper starts with a section about conventional RL and approximate policy iteration, followed by a section about preference-based RL and approximate policy iteration based on label ranking. Our new approach, approximate policy iteration based on dyad ranking, is then introduced in Section 4. In order to confirm the usefulness of the approach, several simulation studies are presented in Section 5, prior to concluding the paper in Section 6.

[1] University of Marburg, Germany, email: dirk.schaefer@uni-marburg.de
[2] Paderborn University, Germany, email:eyke@upb.de

2 REINFORCEMENT LEARNING

Conventional reinforcement learning assumes a scenario in which an agent moves through a (finite) *state space* S by repeatedly selecting *actions* from a set $A = \{\boldsymbol{a}_1, \ldots, \boldsymbol{a}_k\}$. A Markovian *state transition* function $\delta : S \times A \longrightarrow \mathbb{P}(S)$, where $\mathbb{P}(S)$ denotes the set of probability distributions over S, randomly takes the agent to a new state, depending on the current state and the chosen action. Occasionally, the agent receives feedback about its actions in the form of a reward signal $r : S \times A \longrightarrow \mathbb{R}$, where $r(\boldsymbol{s}, \boldsymbol{a})$ is the reward the agent receives for performing action $\boldsymbol{a}$ in state $\boldsymbol{s}$. The goal of the agent is to choose its actions so as to maximize its expected total reward.

The most common task is to learn a *policy* $\pi : S \longrightarrow A$ that prescribes the agent how to act optimally in each situation (state). More specifically, the goal is often defined as maximizing the expected sum of rewards (given the initial state $\boldsymbol{s}$), with future rewards being discounted by a factor $\gamma \in [0, 1]$:

$$V^{\pi}(\boldsymbol{s}) \;=\; E\left[\sum_{t=0}^{\infty} \gamma^t r(\boldsymbol{s}_t, \pi(\boldsymbol{s}_t)) \,|\, \boldsymbol{s}_0 = \boldsymbol{s}\right] \tag{1}$$

where $(\boldsymbol{s}_0, \boldsymbol{s}_1, \boldsymbol{s}_2, \ldots)$ is a trajectory of π through the state space. With $V^*(\boldsymbol{s})$ the best possible value that can be achieved for (1), a policy is called optimal if it achieves the best value in each state $\boldsymbol{s}$. Thus, one possibility to learn an optimal policy is to learn an evaluation of states in the form of a value function [12], or to learn a so-called Q-function which returns the expected reward for a given state-action pair [16]:

$$Q^{\pi}(\boldsymbol{s}, \boldsymbol{a}) = r(\boldsymbol{s}, \boldsymbol{a}) + \gamma \cdot V^{\pi}(\delta(\boldsymbol{s}, \boldsymbol{a}))$$

2.1 Approximate Policy Iteration

Instead of determining optimal actions *indirectly* through learning the value function or the Q-function, one may try to learn a policy *directly* in the form of a mapping from states to actions. A particularly interesting approach in this regard is *approximate policy iteration* (API) with roll-outs [6, 3]. The key idea of this approach is to use a generative model of the underlying process to perform simulations that in turn allow for approximating the value of an action $\boldsymbol{a}$ in a given state $\boldsymbol{s}$. To this end, the action is performed, resulting in a

state $\boldsymbol{s}_1 = \delta(\boldsymbol{s}, \boldsymbol{a})$. The value of this state is estimated by performing so-called *roll-outs*, i.e., by repeatedly selecting actions following a policy π for at most T steps, and finally accumulating the observed rewards. This is repeated several times, and the average reward over the roll-outs is returned as an approximate Q-value $\tilde{Q}^{\pi}(\boldsymbol{s}, \boldsymbol{a})$ for taking action $\boldsymbol{a}$ in state $\boldsymbol{s}$ (leading to $\boldsymbol{s}_1$) and following policy π thereafter.

The roll-outs are then used in a policy iteration loop, which iterates through each of the sample states, simulates all actions in a state, and determines the action $\boldsymbol{a}^*$ that promises the highest Q-value. If $\boldsymbol{a}^*$ is significantly better than all alternative actions in this state, a training example $(\boldsymbol{s}, \boldsymbol{a}^*)$ is added to a training set $\mathcal{T}$, suggesting that $\boldsymbol{a}^*$ is the best action to take in state $\boldsymbol{s}$. Eventually, $\mathcal{T}$ is used for a *policy generalization* step, i.e., to induce a state-action mapping $S \longrightarrow A$ that forms the new policy π'; the underlying problem to be solved is a standard (multi-class) classification problem. This process is repeated several times, until some stopping criterion is met (e.g., if the policy does not improve from one iteration to the next).

3 PREFERENCE-BASED REINFORCEMENT LEARNING

The key idea of *preference-based reinforcement learning* (PBRL) is to replace the (quantitative) evaluation of individual actions by the (qualitative) comparison between pairs of actions [2, 4]. Comparisons of that kind are in principle enough to make optimal decisions. Besides, they are often more natural and less difficult to acquire, especially in applications where the environment does not provide numerical rewards in a natural way.

The basic piece of information we consider is a pairwise preference of the form $\boldsymbol{a}_i \succ_{\boldsymbol{s}} \boldsymbol{a}_j$ or, more specifically, $\boldsymbol{a}_i \succ_{\boldsymbol{s}}^{\pi} \boldsymbol{a}_j$, suggesting that in state $\boldsymbol{s}$, taking action $\boldsymbol{a}_i$ (and following policy π afterwards) is better than taking action $\boldsymbol{a}_j$. Evaluating a trajectory $t = (\boldsymbol{s}_0, \boldsymbol{s}_1, \boldsymbol{s}_2, \ldots)$ in terms of its (expected) total reward (1) reduces the comparison of trajectories to the comparison of real numbers; thus, comparability is enforced and a total order on trajectories is induced. More generally, and arguably more in line with the idea of qualitative feedback, one may assume a partial order relation $\sqsupset$ on trajectories, which means that trajectories t and t' can also be incomparable. A contextual preference can then be defined as follows:

$$\boldsymbol{a}_i \succ_{\boldsymbol{s}}^{\pi} \boldsymbol{a}_j \quad \Leftrightarrow \quad \mathbf{P}(t(\boldsymbol{a}_i) \sqsupset t(\boldsymbol{a}_j)) > \mathbf{P}(t(\boldsymbol{a}_j) \sqsupset t(\boldsymbol{a}_i)) \ , \quad (2)$$

where $t(\boldsymbol{a}_i)$ denotes the (random) trajectory produced by taking action $\boldsymbol{a}_i$ in state $\boldsymbol{s}$ and following π thereafter, and $\mathbf{P}(t \sqsupset t')$ is the probability that trajectory t is preferred to t'. As an in-depth discussion of PBRL is beyond the scope of this paper, we refer the reader to [4] for more details.

3.1 API with Label Ranking

In [4], preference-based reinforcement learning is realized in the form of a preference-based variant of API, namely a variant in which, instead of a classifier $S \longrightarrow A$, a so-called *label ranker* is trained for policy generalization. In the problem of label ranking, the goal is to learn a model that maps instances to rankings over a finite set of predefined choice alternatives [15]. In the context of PBRL, the instance space is given by the state space S, and the set of labels corresponds to the set of actions A. Thus, the goal is to learn a mapping

$$S \longrightarrow \Pi(A) \, ,$$

which maps states to total orders (permutations) of the available actions A. In other words, the task is to learn a function that is able to rank all available actions in a state according to their preference (2).

More concretely, a method called *ranking by pairwise comparison* (RPC) is used for training a label ranker [5]. RPC accepts training information in the form of binary (action) preferences $(\boldsymbol{s}, \boldsymbol{a}_k \succ \boldsymbol{a}_j)$, indicating that in state $\boldsymbol{s}$, action $\boldsymbol{a}_k$ is preferred to action $\boldsymbol{a}_j$. Information of that kind can be produced thanks to the assumption of a generative model as described in Section 2.1. Subsequently, we refer to this approach as API-LR.

4 PBRL USING DYAD RANKING

In comparison to the original, classification-based approach to approximate policy iteration (Section 2.1), the ranking-based method outlined in Section 3.1 exhibits several advantages. For example, pairwise preferences are much easier to elicit for training than examples for unique optimal actions $\boldsymbol{a}^*$. Besides, the preference-based approach better exploits the gathered training information; for example, it utilizes pairwise comparisons between two actions even if both ot them are suboptimal.

In both approaches, however, actions $\boldsymbol{a}_i$ are treated as distinct elements, with no relation to each other; indeed, neither classification nor label ranking do consider any structure on the set of classes A (apart from the trivial discrete structure). Yet, if classes are actions in the context of RL, A is often equipped with a non-trivial structure, because actions can be described in terms of properties/features and can be more or less similar to each other. For example, if an action is an acceleration in a certain direction, like in the mountain car problem (see Section 5 below), then "fast to the right" is obviously more similar to "slowly to the right" than to "fast to the left".

Needless to say, the exploitation of feature-descriptions of actions is a possible way to improve learning in (preference-based) RL, and to generalize, not only over the state space S but also over the action space A. It may allow, for example, to predict the usefulness of actions that have never been tried before. To realize this idea, we make use of so-called *dyad ranking*, a generalization of label ranking that is able to exploit feature-descriptions of labels [9, 10].

4.1 Dyad Ranking

Formally, a dyad is a pair of feature vectors $\boldsymbol{z} = (\boldsymbol{x}, \boldsymbol{y}) \in \mathbb{Z} = \mathbb{X} \times \mathbb{Y}$, where the feature vectors are from two (not necessarily different) domains $\mathbb{X}$ and $\mathbb{Y}$. An example for a dyad is a user - item pair within a recommendation system or a pair of (dis)similar images at a metric learning task. A single training observation ρ_n $(1 \leq n \leq N)$ takes the form of a dyad ranking

$$\rho_n : \boldsymbol{z}^{(1)} \succ \boldsymbol{z}^{(2)} \succ \ldots \succ \boldsymbol{z}^{(M_n)}, \ \ M_n \geq 2, \quad (3)$$

of length M_n which can vary between observations in the data set $\mathcal{D} = \{\rho_n\}_{n=1}^{N}$. The task of a dyad ranking method is to learn a ranking function that accepts as input any set of (new) dyads and produces as output a ranking of these dyads.

An important special case, called *contextual dyad ranking*, is closely related to label ranking [9]. As already mentioned, the label ranking problem is about learning a model that maps instances to rankings over a finite set of predefined choice alternatives. In terms of dyad ranking this means that all dyads in an observation share the same context $\boldsymbol{x}$, i.e., they are all of the form $\boldsymbol{z}^{(j)} = (\boldsymbol{x}, \boldsymbol{y}^{(j)})$; in this case, (3) can also be written as

$$\rho_n : \left(\boldsymbol{x}, \boldsymbol{y}^{(1)}\right) \succ \left(\boldsymbol{x}, \boldsymbol{y}^{(2)}\right) \succ \ldots \succ \left(\boldsymbol{x}, \boldsymbol{y}^{(M_n)}\right) \ . \quad (4)$$

Likewise, a prediction problem will typically consist of ranking a subset

$$\left\{\boldsymbol{y}^{(1)}, \boldsymbol{y}^{(2)}, \ldots, \boldsymbol{y}^{(M)}\right\} \subseteq \mathbb{Y}$$

in a given context $\boldsymbol{x}$. Going back to the example from above, a contextual dyad ranking could be considered as the preferences a user has over a set of items.

4.2 Bilinear Plackett-Luce Model

The Plackett-Luce (PL) model is a statistical model for rank data. Given a set of alternatives $o_1, \ldots, o_K$, it represents a parameterized probability distribution on the set of all rankings over the alternatives. The model is specified by a parameter vector $\boldsymbol{v} = (v_1, v_2, \ldots v_K) \in \mathbb{R}_+^K$, in which v_i accounts for the "strength" of the option o_i. The probability assigned by the PL model to a ranking is represented by a permutation π, where $\pi(i)$ is the index of the option put on position i, is given by

$$\mathbf{P}(\pi \,|\, \boldsymbol{v}) \;=\; \prod_{i=1}^{K} \frac{v_{\pi(i)}}{v_{\pi(i)} + v_{\pi(i+1)} + \ldots + v_{\pi(K)}} \,. \tag{5}$$

In dyad ranking, the options o_i to be ranked are dyads $\boldsymbol{z} = (\boldsymbol{x}, \boldsymbol{y})$. Thus, a model suitable for dyad ranking can be obtained by specifying the PL parameters as a function of the feature vectors $\boldsymbol{x}$ and $\boldsymbol{y}$ [9]:

$$v(\boldsymbol{z}) = v(\boldsymbol{x}, \boldsymbol{y}) = \exp\big(\, \langle \boldsymbol{w}, \Phi(\boldsymbol{x}, \boldsymbol{y}) \rangle \,\big) \;\;, \tag{6}$$

where Φ is a joint feature map [14]. A common choice for such a feature map is the Kronecker product:

$$\Phi(\boldsymbol{x}, \boldsymbol{y}) = \boldsymbol{x} \otimes \boldsymbol{y} = \big(x_1 \cdot y_1, x_1 \cdot y_2, \ldots, x_r \cdot y_c\big) \,, \tag{7}$$

which is a vector consisting of all pairwise products of the components of $\boldsymbol{x}$ and $\boldsymbol{y}$. The equation (7) can equivalently be rewritten as a bilinear form $\boldsymbol{x}^\top \mathbf{W} \boldsymbol{y}$ with a matrix $\mathbf{W} = (w_{i,j})$; the entry $w_{i,j}$ can be considered as the weight of the interaction term $x_i y_j$. This choice of the joint-feature map yields the bilinear version of the PL model:

$$v(\boldsymbol{z}) = v(\boldsymbol{x}, \boldsymbol{y}) = \exp\big(\boldsymbol{x}^\top \mathbf{W} \boldsymbol{y}\big) \;\;, \tag{8}$$

where no constraints on the rectangular matrix $\boldsymbol{W}$ such as positive (semi) definiteness are imposed. Given a set of training data in the form of a set of dyad rankings (3), the learning task comes down to estimating the weight matrix $\mathbf{W}$. Thanks to the probabilistic nature of the model, this can be accomplished by leveraging the principle of maximum likelihood; for details of this approach, we refer to [9].

4.3 API with Dyad Ranking

We are now ready to introduce approximate policy iteration based on dyad ranking (API-DR) as a generalization of API-LR. The former is quite similar to the latter, except that a dyad ranker is trained instead of a label ranker. To this end, training data is again produced by executing a number of rollouts on states, starting with a specified action and following the current policy; see Algorithm 1.

In addition to the representation of actions in terms of features, API-DR has another important advantage. Thanks to the use of the (bilinear) PL model, it is not only able to predict a presumably best action in each state, but also informs about the degree of confidence in that prediction. More specifically, it provides a complete probability distribution over all rankings of actions in each state. Information of this kind is useful for various purposes, as will be discussed next.

Algorithm 1 Approximate Policy Iteration based on Dyad Ranking

Require: sample states $\mathcal{S}$, initial (random) policy π_0, max. number of policy iterations p, subroutine **Evaluate Dyad Ranking** for determining dyad rankings for a given state and a set of permissible actions in that state.

1: **function** API-DR($\mathcal{S}, \pi_0, p$)
2: $\pi \leftarrow \pi_0$, $i \leftarrow 0$
3: **repeat**
4: $\pi' \leftarrow \pi$, $\mathcal{D} \leftarrow \emptyset$
5: **for all** $s \in \mathcal{S}$ **do**
6: $\rho_s \leftarrow$**Evaluate Dyad Ranking** ($\mathcal{A}(s), \pi$)
7: $\mathcal{D} \leftarrow \mathcal{D} \cup \{\rho_s\}$
8: **end for**
9: $\pi \leftarrow$ **Train Dyad Ranker** ($\mathcal{D}$), $i \leftarrow i + 1$
10: **until Stopping Criterion** (π, p)
11: **return** π
12: **end function**

Algorithm 2 Probabilistic Rollout Procedure

Require: Initial state s_0, initial action a_0, policy π, discount factor γ, number of rollouts K, max. length(horizon) of each trajectory L, generative environment model E

1: **function** ROLLOUT(π, s_0, a_0, K, L)
2: **for** $k \leftarrow 1$ to K **do**
3: **while** t $< L$ and $\neg$**Terminal State**(s_{t-1}) **do**
4: $(s_t, r_t) \leftarrow$ **Simulate**(E, s_{t-1}, a_{t-1})
5: $(a_t, p_t) \leftarrow$ **Utilize Policy**(π, s_t)
6: $\widetilde{Q}_k \leftarrow \widetilde{Q}_k + \gamma^t r_t$
7: $t \leftarrow t + 1$
8: **end while**
9: // Remaining rollouts can be skipped if p_t-values are high.
10: **end for**
11: $\widetilde{Q} \leftarrow \frac{1}{k} \sum_{i=1}^{k} \widetilde{Q}_i$
12: **return** $\widetilde{Q}$
13: **end function**

4.3.1 Exploration versus Exploitation

The rollout procedure (Algorithm 2) is invoked by the subroutine *Evaluate Dyad Ranking* (line 6 of Algorithm 1). And there the PL model is used in its role as a policy, which means that it has to prescribe a single action $\boldsymbol{a}^*$ for each state $\boldsymbol{s}$. The most obvious approach is to compute, for each action $\boldsymbol{a}$, the probability

$$\mathbf{P}(\boldsymbol{a} \,|\, \boldsymbol{W}, \boldsymbol{s}) = \frac{\exp(\boldsymbol{s}^\top \boldsymbol{W} \boldsymbol{a})}{\sum_{i=1}^{K} \exp(\boldsymbol{s}^\top \boldsymbol{W} \boldsymbol{a})} \tag{9}$$

of being ranked first, and to choose the action maximizing this probability.

Adopting the presumably best action in each state corresponds to pure *exploitation*. It is well known, however, that successful learning requires a proper balance between *exploration* and exploitation. Interestingly, our approach suggests a very natural way of realizing such a balance, simply by selecting each action $\boldsymbol{a}$ according to its probability (9).

As an aside, we note that a generalization of the PL model can be

used to control the degree of exploration in a more flexible way:

$$\mathbf{P}(\boldsymbol{a} \mid \boldsymbol{W}, \boldsymbol{s}) = \frac{\exp(c \cdot \boldsymbol{s}^\top \boldsymbol{W} \boldsymbol{a})}{\sum_{i=1}^{K} \exp(c \cdot \boldsymbol{s}^\top \boldsymbol{W} \boldsymbol{a})} \tag{10}$$

for a constant $c \geq 0$; the larger c, the stronger the strategy focuses on the best actions.

4.3.2 *Uncertainty Sampling*

Another interesting opportunity to exploit probabilistic information is for *active learning* via *uncertainty sampling*. Uncertainty sampling is a general strategy for active learning in which those training examples are requested for which the learner appears to be maximally uncertain [11]. In binary classification, for example, these are typically the instances that are located closest to the (current) decision boundary.

In our case, the distribution (9) informs about the certainty or uncertainty of the learner regarding the best course of action in a given state $\boldsymbol{s}$ (or, alternatively, the uncertainty about the true ranking of all actions in that state). This uncertainty can be quantified, for instance, in terms of the entropy of that distribution, or the margin between the probability of the best and the second-best action. Correspondingly, those states can be selected as sample states $\mathcal{S}$ in Algorithm 1 for which the uncertainty is highest.

5 EXPERIMENTS

5.1 Standard Benchmarks

5.1.1 *Inverted Pendulum*

The inverted pendulum (also known as *cart pole*) problem (IP) is to balance a pendulum which is attached on top of a cart. The only way to stabilize the pendulum is by moving the cart, which is placed on a planar ground, to the left or to the right. We adopt the experimental setting from Lagoudakis and Parr [6], in which the position of the cart in space is not taken into account (see Figure 1 (a)).

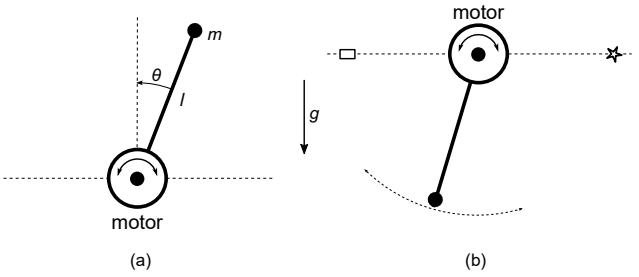


Figure 1. Schematic diagrams of (a) the inverted pendulum and (b) the mountain car problem. In (a) the goal is to keep the pendulum close to the vertical axis for a period of time, whereas in (b) an underpowered motor must be used in interaction with the gravity to reach the star.

In the original formulation, there are three actions possible which are mapped, respectively, onto the forces of $\{-10, 0, 10\}$ Newtons. The state space is continuous and two-dimensional. The first dimension captures the angle θ between the pole and the vertical axis, whereas the second dimension describes the angle velocity $\dot{\theta}$. The transitions of the physical model are determined by the nonlinear dynamics of the system; they depend on the current state $s = (\theta, \dot{\theta})$ and the current action value a, respectively:

$$\ddot{\theta} = \frac{g \sin(\theta) - \alpha m l (\dot{\theta})^2 \sin(2\theta)/2 - \alpha \cos(\theta) a}{4l/3 - \alpha m l \cos^2(\theta)} ,$$

where $\alpha = 1/(m + M)$ and the residual parameters are chosen as in Lagoudakis and Parr (see Table 1).

Parameter	Symbol	Value	Unit
Gravity	g	9.81	m/s^2
Cart mass	M	8.0	kg
Pendulum mass	m	2.0	kg
Pendulum length	l	0.5	m

Table 1. Inverted pendulum model parameters.

5.1.2 *Mountain Car*

The mountain car problem (MC) consists of driving an underpowered car out of a valley (see Figure 1 (b) for a schematic diagram). The agent must learn a policy which takes the momentum of the car into account when driving the car along the valley sides. It can basically power or throttle forwards and backwards. At each time step, the system dynamics depend on a state $s_t = (x_t, \dot{x}_t)$ and an action a_t. It is described by the following equations:

$$x_{t+1} = b_1(x_t + \dot{x}_{t+1})$$
$$\dot{x}_{t+1} = b_2(\dot{x}_t + 0.001 a_t - 0.025 \cos(3x_t)),$$

where b_1 is a function that restricts the position x to the interval $[-1.2, 0.5]$ and b_2 restricts the velocity to the interval $[-0.07, 0.07]$. In case the agent reaches $x_t = -1.2$, an inelastic collision is simulated by setting the velocity $\dot{x}$ to zero. The gravity depends on the local slope of the mountain, which is simulated with the term $0.025 \cos(3x_t)$. As long as the position x is less then 0.5, the agent receives zero reward. If the car hits the right bound ($x = 0.5$), the goal is achieved, the episode ends, and the agent obtains reward 1.

In both problems, the actions are simulated to be noisy, which results in non-deterministic state transitions. Thus, the learner is required to perform multiple rollouts. In particular, we add random noise from the intervals $[-0.2, 0.2]$ and $[-0.01, 0.01]$ to the raw action signals for IP and MC, respectively.

5.1.3 *Experiment*

For evaluation, we follow [3, 2] by plotting the cumulative distribution of success rates over a measure of complexity, i.e. the number of actions needed throughout the API procedure for generating a policy that solves a task successfully. In the case of the MC/IP tasks the number is obtained by summing up the average numbers of actions performed for each of the K rollouts realized on initial (state, action) pairs[3]. A point (x, y) in these plots can be interpreted as the minimum number of actions x required to reach a success rate of y.

[3] Note that the number of actions is not fixed per rollout and rather depends on the quality of the current policy. This includes the case that rollouts can stop prematurely before the maximal trajectory length L is reached.

We hypothesize that the incorporation of action features can improve the quality of learned policies, especially in situations where data is scarse. To this end, the quality of policies learned by API-LR[4] and API-DR are measured under different conditions. We chose a moderate number of 17 actions on both environments by dividing the original number range into 17 equally sized parts. We furthermore defined three conditions referred to as *complete*, *partial* and *duel*. Under the first condition, preferences about the entire action set are available per state. In the *partial* condition, the learner can only learn from three randomly drawn actions per state. In the last condition, only two actions are drawn, leading to only one preference per state. Under all condition the number of sampled states $|\mathcal{S}|$ were set fixed to 50 for the MC task and 100 for the IP task. The results depicted in Figure 2 clearly confirm our expectations.

5.2 Cancer Clinical Trials Simulation

Preference-based reinforcement learning has been specifically motivated by the example of optimal therapy design in cancer treatment [2]. Here, it is more natural to work with preferences instead of numerical reward functions, because in a clinical context, it is often very difficult to specify the health state (including extreme events such as death) of a patient numerically. Another characteristic of the cancer treatment scenario is the evaluation of therapy plans on *multiple* criteria [7], leading to incomparability and preference relations in terms of partial orders.

The scenario we consider here is based on a mathematical model of [17] that captures the tumor growth during a treatment, the level of toxicity (inversely related to the wellness of the patient) due to the chemotherapy, the effect of the treatment and the interaction between drug and tumor size. A state is described by the variables tumor size S and toxicity X, while actions correspond to the dosage level $D \in [0, 1]$ of the drug. The model is described by a system of difference equations $S_{t+1} = S_t + \Delta S_t$ and $X_{t+1} = X_t + \Delta X_t$, where

$$\Delta S_t = \Big(a_1 \cdot \max(X_t, X_0) - b_1 \cdot (D_t - d_1)\Big) \cdot \mathbb{1}_{S_t > 0},$$
$$\Delta X_t = a_2 \cdot \max(S_t, S_0) + b_2 \cdot (D_t - d_2).$$

The probability for a patient to die in the t-th month follows a Bernoulli distribution with parameter $p = 1 - \exp(-\gamma(t))$ using the hazard function $\log \gamma(t) = c_0 + c_1 S_t + c_2 X_t$. Following the recommendation of [17], we fix the parameters of the difference equation as follows: $a_1 = 0.15$, $a_2 = 0.1$, $b_1 = b_2 = 1.2$, $d_1 = d_2 = 0.5$ and $c_0 = -4$, $c_1 = c_2 = 0.5$. The problem is how to choose appropriate dosage levels during a therapy of 6 months.

To circumvent the problem of reward function specification, we propose a preference-based comparison of policies π and π' as follows: π is preferred to π' if a patient survives with policy π and dies under π'. If a patient does not survive under either of the policies, then these are considered to be incomparable. If a patient survives under both policies, we define the preference via Pareto dominance as

$$\pi \succeq \pi' \Leftrightarrow (C_X \leq C'_X) \text{ and } (C_S \leq C'_S) \ , \qquad (11)$$

in which C_X denotes the maximal toxicity level occurred within a 6 month treatment under policy π, and analogously C'_X for π'. C_S and C'_S denote the tumor sizes at the end of the therapy after corresponding to policies π and π' respectively.

[4] Throughout all experiments we used the RPC method in conjunction with logistic regression.

5.2.1 Experiment

We applied API-DR with the feature representation $\boldsymbol{x} = (1, S, X)$ and $\boldsymbol{y} = (1, D, D^2, D^3)$. Moreover, the experimental protocol follows that of [2], in which virtual patients were generated by sampling initial states independently and uniformly from the interval $(0, 2)$. For training, 1000 patients were taken, and the quality of the learned policies was then tested on 200 new patients. In addition to API-LR, we also included a random policy and several constant policies als baselines. While the former selects dosages uniformly at random, these latter always prescribe the same dosage level regardless the patient's health state: extreme (1.0), high (0.7), medium (0.4) and low (0.1). This division of 4 dosages has also been used as the set of available actions. In contrast to API-LR which utilizes the labels *extreme*, *high*, *medium* and *low*, API-DR is able to utilize their associated numerical values.

Since the objective is to perform strongly on all three criteria, i.e., tumor size, toxicity level, and death rate, the performance is shown in three plots (see Figure 3), one for each pair of criteria. As can be seen, API-DR has advantages in comparison to the other approaches in all aspects: final tumor size, average toxicity level, and the death rate. In comparison with the constant approaches, API-DR is worse than the extreme constant dosage level in terms of final tumor size but superior in terms of death rate and toxicity levels.

6 CONCLUSION

We proposed a combination of (preference-based) reinforcement learning and dyad ranking that is applicable in situations where qualitative instead of quantitative preference information on state-action trajectories is available. This setting extends the existing preference-based reinforcement learning framework by incorporating feature descriptions of actions and considering rankings of dyads, i.e., state/action pairs, instead of rankings of actions given states. Thus, it becomes possible to generalize over the state and the action space simultaneously.

Building on our recent work [8], we plan to generalize our approach further by using neural networks instead of a bilinear function as a joint feature map in the Plackett-Luce model. While the assumption of (bi)linearity is often too restrictive, neural networks allow for capturing nonlinear interactions between state and action features. Moreover, we plan to elaborate on the ideas for exploiting probabilistic information as outlined in Section 4.3.

REFERENCES

[1] Riad Akrour, Marc Schoenauer, and Michele Sebag, 'Preference-based policy learning', in *Proceedings ECML/PKDD–2011, European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases*, Athens, Greece, (2011).

[2] W. Cheng, J. Fürnkranz, E. Hüllermeier, and S.H. Park, 'Preference-based policy iteration: Leveraging preference learning for reinforcement learning', in *Proceedings ECML/PKDD–2011, European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases*, Athens, Greece, (2011).

[3] Christos Dimitrakakis and Michail G Lagoudakis, 'Rollout sampling approximate policy iteration', *Machine Learning*, **72**(3), 157–171, (2008).

[4] Johannes Fürnkranz, Eyke Hüllermeier, Weiwei Cheng, and Sang-Hyeun Park, 'Preference-based reinforcement learning: a formal framework and a policy iteration algorithm', *Machine Learning*, **89**(1-2), 123–156, (2012).

[5] E. Hüllermeier, J. Fürnkranz, W. Cheng, and K. Brinker, 'Label ranking by learning pairwise preferences', *Artificial Intelligence*, **172**, 1897–1917, (2008).

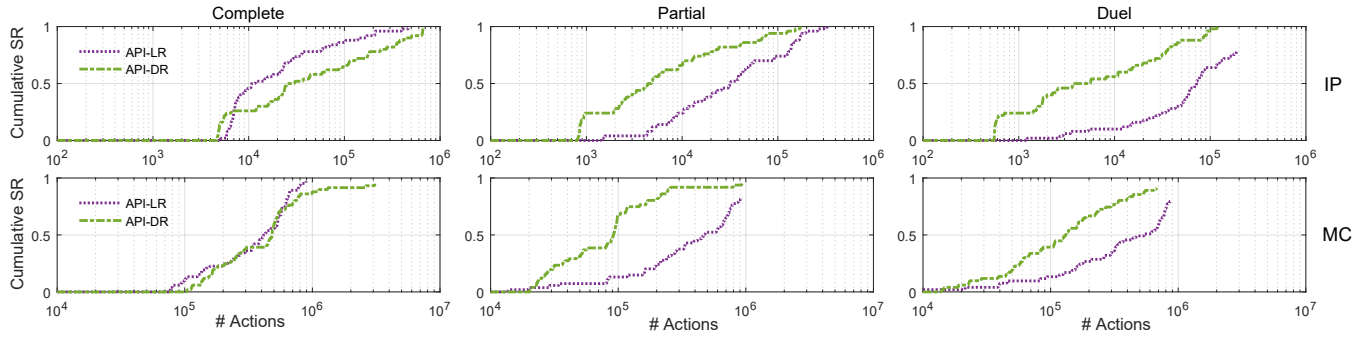


Figure 2. Performance of the methods for the inverted pendulum (first row) and the mountain car task (second row).

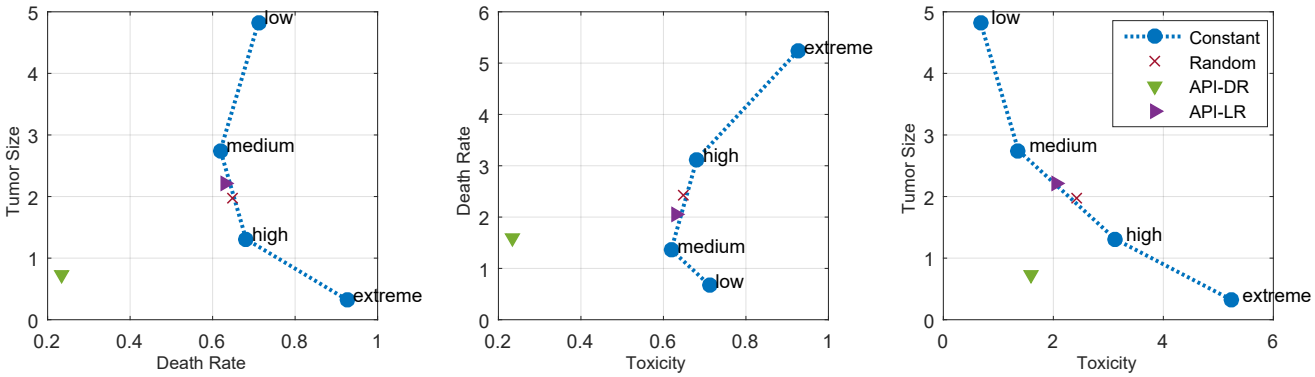


Figure 3. Results of the cancer clinical trials simulation.

[6] M. Lagoudakis and R. Parr, 'Reinforcement learning as classification: Leveraging modern classifiers', in *Proceedings ICML, 20th International Conference on Machine Learning*, eds., Tom Fawcett and Nina Mishra, volume 20, pp. 424–431. AAAI Press, (2003).

[7] Bernard Roy, 'Decision-aid and decision-making', *European Journal of Operational Research*, **45**(2-3), 324–331, (1990).

[8] D. Schäfer and E. Hüllermeier, 'Plackett-Luce networks for dyad ranking', in *Workshop LWDA, "Lernen, Wissen, Daten, Analysen"*, Potsdam, Germany, (2016).

[9] Dirk Schäfer and Eyke Hüllermeier, 'Dyad ranking using a bilinear Plackett-Luce model', in *Proceedings ECML/PKDD–2015, European Conference on Machine Learning and Knowledge Discovery in Databases*, Porto, Portugal, (2015). Springer.

[10] Dirk Schäfer and Eyke Hüllermeier, 'Preference-based meta-learning using dyad ranking: Recommending algorithms in cold-start situations.', in *MetaSel@PKDD/ECML*, eds., Joaquin Vanschoren, Pavel Brazdil, Christophe G. Giraud-Carrier, and Lars Kotthoff, volume 1455 of *CEUR Workshop Proceedings*, pp. 110–111. CEUR-WS.org, (2015).

[11] Burr Settles, 'Active learning literature survey', Technical Report 1648, University of Wisconsin-Madison, (2008).

[12] Richard S. Sutton, 'Learning to predict by the methods of temporal differences', *Machine Learning*, **3**(1), 9–44, (1988).

[13] R.S. Sutton and A.G. Barto, *Reinforcement Learning: An Introduction*, MIT Press, Cambridge, MA, 1998.

[14] Ioannis Tsochantaridis, Thorsten Joachims, Thomas Hofmann, and Yasemin Altun, 'Large margin methods for structured and interdependent output variables', in *Journal of Machine Learning Research*, volume 6, pp. 1453–1484. JMLR.org, (2005).

[15] S. Vembu and T. Gärtner, 'Label ranking: a survey', in *Preference Learning*, eds., J. Fürnkranz and E. Hüllermeier, Springer, (2010).

[16] Christopher J. C. H. Watkins and Peter Dayan, 'Q-learning', *Machine Learning*, **8**(3), 272–292, (1992).

[17] Zhao, Osorok, and Zeng, 'Reinforcement learning design for cancer clinical trials', *Statistics in medicine*, **28**(15), 1982–1998, (2009).

Learning MR-Sort rules with coalitional veto

Olivier Sobrie[1,2,3] **, Vincent Mousseau**[2] and **Marc Pirlot**[3]

Abstract. MR-Sort (Majority Rule Sorting) is a multiple criteria sorting method which assigns an alternative a to category C^h when a is better than the lower limit of C^h on a majority of criteria, and this is not true with the upper limit of C^h. We enrich the descriptive ability of MR-Sort by the addition of coalitional vetoes which operate in a symmetric way as compared to the MR-Sort rule w.r.t. to category limits, using specific veto profiles and veto weights. We describe a heuristic algorithm to learn such an MR-Sort model enriched with coalitional veto from a set of assignment examples, and show how it performs on real datasets.

1 Introduction

Multiple Criteria Sorting Problems aim at assigning alternatives to one of the predefined ordered categories $C^1, C^2, ..., C^p$, C^1 and C^p being the worst and the best category, respectively. Many multiple criteria sorting methods have been proposed in the literature (see e.g., [9], [25]). MR-Sort (Majority Rule Sorting, see [10]) is an outranking-based multiple criteria sorting method which corresponds to a simplified version of ELECTRE TRI where the discrimination and veto thresholds are omitted.

In the pessimistic version of ELECTRE TRI, veto effects make it possible to worsen the category to which an alternative is assigned when this alternative has very bad performances on one/several criteria. We consider a variant of MR-Sort which introduces possible veto effects. While in ELECTRE TRI, a veto involves a single criterion, we consider a more general formulation of veto (see [21]) which can involve a coalition of criteria (such a coalition can be reduced to a singleton).

The definition of such a "*coalitional veto*" exhibits a noteworthy symmetry between veto and concordance. To put it simple, in a two-category context (*Bad/Good*), an alternative is classified as *Good* when its performances are above the concordance profile on a sufficient majority of criteria, and when its performances are not below the veto profile for a sufficient majority of criteria. Hence, the veto condition can be viewed as the negation of a majority rule using a specific veto profile, and specific veto weights.

Algorithms to learn the parameters of an MR-Sort model without veto (category limits and criteria weights) have been proposed, either using linear programming involving integer variables (see [10]) or using a specific heuristic (see [20, 19]). When the size of the learning set exceeds 100, only heuristic algorithms are able to provide a solution within a limited computing time.

Olteanu and Meyer [14] have developed a simulated annealing based algorithm to learn a MR-Sort model with classical veto (not coalitional ones).

In this paper, we propose a new heuristic algorithm to learn the parameters of a MR-Sort model with coalitional veto (called MR-Sort-CV) which makes use of the symmetry between the concordance and the coalitional veto conditions. We describe preliminary results obtained by experimenting with this algorithm on real data sets.

The paper is organized as follows. In Section 2, we recall MR-Sort and define its extension when considering monocriterion veto and coalitional veto. After a brief reminder of the heuristic algorithm to learn an MR-Sort model, Section 3 is devoted to the presentation of the algorithm to learn an MR-Sort model with coalitional veto. Section 4 presents experimentations of this algorithm and Section 5 groups conclusions and directions for further research.

2 Considering vetoes in MR-Sort

2.1 MR-Sort model

MR-Sort is a method for assigning objects to ordered categories. It is a simplified version of ELECTRE TRI, another MCDA method [23, 16].

The MR-Sort rule works as follows. Formally, let X be a set of objects evaluated on n ordered attributes (or criteria), $F = \{1, ..., n\}$. We assume that X is the Cartesian product of the criteria scales, $X = \prod_{j=1}^{n} X_j$, each scale X_j being completely ordered by the relation $\geq_j$. An object $a \in X$ is a vector $(a_1, \ldots, a_j, \ldots, a_n)$, where $a_j \in X_j$ for all j. The ordered categories which the objects are assigned to by the MR-Sort model are denoted by C^h, with $h = 1, \ldots, p$. Category C^h is delimited by its lower limit profile b^{h-1} and its upper limit profile b^h, which is also the lower limit profile of category C^{h+1} (provided $0 < h < p$). The profile b^h is the vector of criterion values $(b_1^h, \ldots, b_j^h, \ldots, b_n^h)$, with $b_j^h \in X_j$ for all j. We denote by $P = \{1, ..., p\}$ the list of category indices. By convention, the best category, C^p, is delimited by a fictive upper profile, b^p, and the worst one, C^1, by a fictive lower profile, b^0. It is assumed that the profiles dominate one another, i.e.: $b_j^h \geq_j b_j^{h-1}$, for $h = \{1, \ldots, p\}$ and $j = \{1, \ldots, n\}$.

Using the MR-Sort procedure, an object is assigned to a category if its criterion values are at least as good as the category lower profile values on a weighted majority of criteria while this condition is not fulfilled when the object's criterion

[1] email: olivier.sobrie@gmail.com
[2] CentraleSupélec, Université Paris-Saclay, Grande Voie des Vignes, 92295 Châtenay-Malabry, France, email: vincent.mousseau@centralesupelec.fr
[3] Université de Mons, Faculté Polytechnique, 9, rue de Houdain, 7000 Mons, Belgium, email: marc.pirlot@umons.ac.be

values are compared to the category upper profile values. In the former case, we say that the object is *preferred* to the profile, while, in the latter, it is not. Formally, if an object $a \in X$ is *preferred* to a profile b^h, we denote this by $a \succcurlyeq b^h$. Object a is preferred to profile b^h whenever the following condition is met:

$$a \succcurlyeq b^h \Leftrightarrow \sum_{j:a_j \geq_j b_j^h} w_j \geq \lambda, \tag{1}$$

where w_j is the nonnegative weight associated with criterion j, for all j and λ sets a majority level. The weights satisfy the normalization condition $\sum_{j \in F} w_j = 1$; λ is called the *majority threshold*.

The preference relation $\succcurlyeq$ defined by (1) is called an *outranking* relation without veto or a *concordance* relation ([16]; see also [3, 4] for an axiomatic description of such relations). Consequently, the condition for an object $a \in X$ to be assigned to category C^h reads:

$$\sum_{j:a_j \geq_j b_j^{h-1}} w_j \geq \lambda \quad \text{and} \quad \sum_{j:a_j \geq_j b_j^h} w_j < \lambda. \tag{2}$$

The MR-Sort assignment rule described above involves $pn+1$ parameters, i.e. n weights, $(p-1)n$ profiles evaluations and one majority threshold.

A *learning set* A is a subset of objects $A \subseteq X$ for which an assignment is known. For $h \in P$, A_h denotes the subset of objects $a \in A$ which are assigned to category C^h. The subsets A_h are disjoint; some of them may be empty.

2.2 MR-Sort-MV

In this section, we recall the traditional monocriterion veto rule as defined by [1, 2]. In a MR-Sort model with monocriterion veto, an alternative a is "at least as good as" a profile b^h if it has at least equal to or better performances than b^h on a weighted majority of criteria and if it is not strongly worse than the profile on any criterion. In the sequel, we call b^h a *concordance profile* and we define "strongly worse than the profile" b^h by means of a veto profile $v^h = (v_1^h, v_2^h, ..., v_n^h)$, with $v_j^h \leq_j b_j^h$. It represents a vector of performances such that any alternative having a performance worse than or equal to this profile on any criterion would be excluded from category C^{h+1}. Formally, the assignment rule is described by the following condition:

$$a \succcurlyeq b^h \iff \sum_{j:a_j \geq_j b_j^h} w_j \geq \lambda \text{ and not } a\, V b^h,$$

with $a\, V b^h \iff \exists j \in F : a_j \leq_j v_j^h$. Note that non-veto condition is frequently presented in the literature using a veto threshold (see e.g. [15]), i.e. a maximal difference w.r.t. the concordance profile in order to be assigned to the category above the profile. Using veto profiles instead of veto thresholds better suits the context of multicriteria sorting. We recall that a profile b^h delimits the category C^h from C^{h+1}, with $C^{h+1} \succ C^h$; with monocriterion veto, the MR-Sort assignment rule reads as follows:

$$a \in C^h \iff \left[\sum_{j:a_j \geq_j b_j^{h-1}} w_j \geq \lambda \text{ and } \nexists j \in F : a_j < v_j^{h-1} \right] \text{ and } \left[\sum_{j:a_j \geq_j b_j^h} w_j < \lambda \text{ or } \exists j \in F : a_j \leq v_j^h \right]. \tag{3}$$

We remark that a MR-Sort model with more than 2 categories remains consistent only if veto profiles v_j^h do not overlap, i.e, are chosen such that $v_j^h \geq v_j^{h'}$ for all $\{h, h'\}$ s.t. $h > h'$. Otherwise, an alternative might be on the one hand in veto against a profile b^h, which prevents it to be assigned to C^{h+1} and, on the other hand, not in veto against b^{h+1}, which does not prevent it to be assigned to C^{h+2}.

2.3 MR-Sort-CV

We introduce here a new veto rule considering vetoes w.r.t. coalitions of criteria, which we call "*coalitional veto*". With this rule, the veto applies and forbids an alternative a to be assigned to category C^{h+1} when the performance of an alternative a is not better than v_j^h on a weighted majority of criteria.

As for the monocriterion veto, the veto profiles are vectors of performances $v^h = (v_1^h, v_2^h, ..., v_n^h)$, for all $h = \{1, .., p\}$. Coalitional veto also involves a set of *veto weights* denoted z_j, for all $j \in F$. Without loss of generality, the sum of z_j is set to 1. Furthermore, a veto cutting threshold Λ is also involved and determines whether a coalition of criteria is sufficient to impose a veto. Formally, we express the coalitional veto rule $a\, V b^h$, as follows:

$$a\, V b^h \iff \sum_{j:a_j \leq_j v_j^h} z_j \geq \Lambda. \tag{4}$$

Using coalitional veto, the outranking relation of MR-Sort (2.2) is modified as follows:

$$a \succcurlyeq b^h \iff \sum_{j:a_j \geq_j b_j^h} w_j \geq \lambda \text{ and } \sum_{j:a_j \leq_j v_j^h} z_j < \Lambda. \tag{5}$$

Using coalitional veto with MR-Sort modifies the assignment rule as follows:

$$a \in C^h \iff \left[\sum_{j:a_j \geq_j b_j^{h-1}} w_j \geq \lambda \text{ and } \sum_{j:a_j \leq_j v_j^{h-1}} z_j < \Lambda \right] \text{ and } \left[\sum_{j:a_j \geq_j b_j^h} w_j < \lambda \text{ or } \sum_{j:a_j \leq_j v_j^h} z_j \geq \Lambda \right] \tag{6}$$

In MR-Sort, the coalitional veto can be interpreted as a combination of performances preventing the assignment of an alternative to a category. We call this new model, MR-Sort-CV.

The coalitional veto rule given in Equation (5) is a generalization of the monocriterion rule. Indeed, if the veto cut threshold Λ is equal to $\frac{1}{n}$ (n being the number of criteria), and each veto weight z_j is set to $\frac{1}{n}$, then the veto rule defined in Equation (4) corresponds to a monocriterion veto for each criterion.

2.4 The Non Compensatory Sorting (NCS) model

In this subsection, we recall the non compensatory sorting (NCS) rule as defined by [1, 2], which will be used in the experimental part (Section 4) for comparison purposes. These rules allow to model criteria interactions. MR-Sort is a particular case of these, in which criteria do not interact.

In order to take criteria interactions into account, it has been proposed to modify the definition of the global outranking relation, $a \succcurlyeq b^h$, given in (1). We introduce the notion of capacity. A *capacity* is a function $\mu : 2^F \to [0, 1]$ such that:

- $\mu(B) \geq \mu(A)$, for all $A \subseteq B \subseteq F$ (monotonicity) ;
- $\mu(\emptyset) = 0$ and $\mu(F) = 1$ (normalization).

The Möbius transform allows to express the capacity in another form:

$$\mu(A) = \sum_{B \subseteq A} m(B), \tag{7}$$

for all $A \subseteq F$, with $m(B)$ defined as:

$$m(B) = \sum_{C \subseteq B} (-1)^{|B|-|C|} \mu(C) \tag{8}$$

The value $m(B)$ can be interpreted as the weight that is exclusively allocated to B as a whole. A capacity can be defined directly by its Möbius transform also called "interaction". An interaction m is a set function $m : 2^F \to [-1, 1]$ satisfying the following conditions:

$$\sum_{j \in K \subseteq J \cup \{j\}} m(K) \geq 0, \quad \forall j \in F, J \subseteq F \backslash \{i\} \tag{9}$$

and

$$\sum_{K \subseteq F} m(K) = 1.$$

If m is an interaction, the set function defined by $\mu(A) = \sum_{B \subseteq A} m(B)$ is a capacity. Conditions (9) guarantee that μ is monotone [6].

Using a capacity to express the weight of the coalition in favor of an object, we transform the outranking rule as follows:

$$a \succcurlyeq b^h \Leftrightarrow \mu(A) \geq \lambda \text{ with } A = \{j : a_j \geq_j b_j^h\} \text{ and } \mu(A) = \sum_{B \subseteq A} m(B) \tag{10}$$

Computing the value of $\mu(A)$ with the Möbius transform induces the evaluation of $2^{|A|}$ parameters. In a model composed of n criteria, it implies the elicitation of 2^n parameters, with $\mu(\emptyset) = 0$ and $\mu(F) = 1$. To reduce the number of parameters to elicit, we use a 2-additive capacity in which all the interactions involving more than 2 criteria are equal to zero. Inferring a 2-additive capacity for a model having n criteria requires the determination of $\frac{n(n+1)}{2} - 1$ parameters.

Finally, the condition for an object $a \in X$ to be assigned to category C^h can be expressed as follows:

$$\mu(F_{a,h-1}) \geq \lambda \quad \text{and} \quad \mu(F_{a,h}) < \lambda \tag{11}$$

with $F_{a,h-1} = \{j : a_j \geq_j b_j^{h-1}\}$ and $F_{a,h} = \{j : a_j \geq_j b_j^h\}$.

3 Learning MR-Sort

Learning the parameters of MR-Sort and ELECTRE TRI models has been already studied in several articles [10, 17, 12, 11, 13, 7, 8, 5, 24]. In this section, we recall how to learn the parameters of an MR-Sort model using respectively an exact method [10] and a heuristic algorithm [17]. We then extend the heuristic algorithm to MR-Sort-CV.

3.1 Learning a simple MR-Sort

It is possible to learn a MR-Sort model from a learning set using Mixed Integer Programming (MIP), see [10]. Such a MIP formulation is not suitable for large data sets because of the high computing time required to infer the MR-Sort parameters. In view of learning MR-Sort models in the context of large data sets, a heuristic algorithm has been proposed in [17]. As for the MIP, the heuristic algorithm takes as input a set of assignment examples and their vectors of performances. The algorithm returns the parameters of a MR-Sort model.

The heuristic algorithm proposed in [17] works as follows. First a population of N_{mod} MR-Sort models is initialized. Thereafter, the following two steps are repeated iteratively on each model in the population:

1. A linear program optimizes the weights and the majority threshold on the basis of assignment examples and fixed profiles.
2. Given the inferred weights and the majority threshold, a heuristic adjusts the profiles of the model on the basis of the assignment examples.

After applying these two steps to all the models in the population, the $\left\lfloor \frac{N_{\text{mod}}}{2} \right\rfloor$ models restoring the least numbers of examples are reinitialized. These steps are repeated until the heuristic finds a model that fully restores all the examples or after a number of iterations specified a priori.

The linear program designed to learn the weights and the majority threshold is given by (12). It minimizes a sum of slack variables, x'_a and y'_a, that is equal to 0 when all the objects are correctly assigned, i.e. assigned to the category defined in the input data set. We remark that the objective function of the linear program does not explicitly minimize the 0/1 loss but a sum of slacks. This implies that compensatory effects might appear, with undesirable consequences on the 0/1 loss. However in this heuristic, we consider that these effects are acceptable. The linear program doesn't involve binary variables. Therefore, the computing time remains reasonable when the size of the problem increases.

The objective function of the heuristic varying the profiles maximizes the number of examples compatible with the model. To do so, it iterates over each profile b^h and each criterion j and identifies a set of candidate moves for the profile, which correspond to the performances of the examples on criterion j located between profiles b^{h-1} and b^{h+1}. Each candidate move is evaluated as a function of the probability to improve the classification accuracy of the model. To evaluate if a candidate move is likely or unlikely to improve the classification of one or several objects, the examples which have an evaluation on criterion j located between the current value of the profile, b_j^h, and the candidate move, $b_j^h + \delta$ (resp. $b_j^h - \delta$), are grouped in different subsets:

$$
\begin{array}{l}
\min \sum_{a \in A}(x'_a + y'_a) \\
\text{s.t.} \\
\quad \sum_{j:a_j \geq_j b_j^{h-1}} w_j - x_a + x'_a = \lambda \qquad \forall a \in A_h, h = \{2, ..., p\} \\
\quad \sum_{j:a_j \geq_j b_j^{h}} w_j + y_a - y'_a = \lambda - \epsilon \qquad \forall a \in A_h, h = \{1, ..., p-1\} \\
\quad \sum_{j=1}^{n} w_j = 1 \\
\quad w_j \in [0;1] \qquad \forall j \in F \\
\quad \lambda \in [0;1] \\
\quad x_a, y_a, x'_a, y'_a \in \mathbb{R}_0^+ \\
\quad \varepsilon \text{ a small positive number.}
\end{array}
\tag{12}
$$

$V_{h,j}^{+\delta}$ **(resp.** $V_{h,j}^{-\delta}$**)** : the sets of objects misclassified in C^{h+1} instead of C^h (resp. C^h instead of C^{h+1}), for which moving the profile b^h by $+\delta$ (resp. $-\delta$) on j results in a correct assignment.

$W_{h,j}^{+\delta}$ **(resp.** $W_{h,j}^{-\delta}$**)** : the sets of objects misclassified in C^{h+1} instead of C^h (resp. C^h instead of C^{h+1}), for which moving the profile b^h by $+\delta$ (resp. $-\delta$) on j strengthens the criteria coalition in favor of the correct classification but will not by itself result in a correct assignment.

$Q_{h,j}^{+\delta}$ **(resp.** $Q_{h,j}^{-\delta}$**)** : the sets of objects correctly classified in C^{h+1} (resp. C^{h+1}) for which moving the profile b^h by $+\delta$ (resp. $-\delta$) on j results in a misclassification.

$R_{h,j}^{+\delta}$ **(resp.** $R_{h,j}^{-\delta}$**)** : the sets of objects misclassified in C^{h+1} instead of C^h (resp. C^h instead of C^{h+1}), for which moving the profile b^h by $+\delta$ (resp. $-\delta$) on j weakens the criteria coalition in favor of the correct classification but does not induce misclassification by itself.

$T_{h,j}^{+\delta}$ **(resp.** $T_{h,j}^{-\delta}$**)** : the sets of objects misclassified in a category higher than C^h (resp. in a category lower than C^{h+1}) for which the current profile evaluation weakens the criteria coalition in favor of the correct classification.

A formal definition of these sets can be found in [17]. The evaluation of the candidate moves is done by aggregating the number of elements in each subset. Finally, the choice to move or not the profile on the criterion is determined by comparing the candidate move evaluation to a random number drawn uniformly. These operations are repeated multiple times on each profile and each criterion.

3.2 Learning MR-Sort-CV

In (2), the MR-Sort condition $\sum_{j:a_j \geq_j b_j^{h-1}} w_j \geq \lambda$ is a necessary condition for an alternative to be assigned to a category at least as good as C^h. Basically a coalitional veto rule can be viewed as a dual version of the majority rule. It provides a sufficient condition for being assigned to a category worse than C^h. An alternative will be assigned to such a category as soon as $\sum_{j:a_j \leq_j v_j^{h-1}} z_j \geq \Lambda$. This condition has essentially the same form as the MR-Sort rule except that the sum is over the criteria on which the alternative's performance is at most as good as the profile (instead of at least as good, in the MR-Sort rule). Therefore, a straightforward way of implementing an algorithm to learn a MR-Sort-CV model is by using the MR-Sort learning heuristic twice, the second time, looking at each criterion in the reversed order of preference.

In the first step, we learn concordance profiles b^h, a weight vector w and a threshold λ using the MR-Sort learning heuristic [18]. We tune the parameters of this algorithm in order to penalize more the false negative than the false positive assignments. In a second step, we apply essentially the same algorithm to learn veto profiles v^h, a weight vector z and a threshold Λ. The direction of optimization is reversed on each criterion, the veto profiles are constrained to lie below their corresponding concordance profile (i.e. $v_j^h \leq_j b_j^h$, for all j and h), which was determined in the first step. In the second step, the learning algorithm is applied to all assignment examples.

We now give more detail on the way we tuned the parameters of the algorithm used in the first step. Let us call the model obtained in the first step, for category C^h, the concordance rule, and the model in the second step, the veto rule. The main point is that the false positive and the false negative assignments produced by the concordance rule are not treated equally. False positive assignments can be corrected by the veto rule, while false negatives cannot. Moreover, the second step leading to a veto rule will have little impact on classification accuracy in case the proportion of false positives is small in the set of wrongly assigned alternatives. For these reasons, we had to penalize more severely false negatives than false positives in the first step. There are basically three simple actions on the algorithm's parameter that can result in favoring false positive assignments.

1. Model selection process. After having iterated the two steps of the algorithm (weights optimization and profiles adjustment) described in Section 3.1, $[N_{\text{mod}}/2]$ models are discarded and replaced. This is done, in the MR-Sort learning algorithm, by selecting the models that make the more assignment errors. We adapt this selection criterion by adding 0.3 times the number of false positive assignments to the total number of correct assignments. The discarded models are thus those for which the number of true positive plus the number of true negative plus 0.3 times the number of false positive is below the median of that quantity on the models' population.
2. Weights optimization. The concordance profiles being given, the weights are optimized using the linear program (12). The sum of the error variables $x'_a + y'_a$ was the objective to be minimized. In the linear program, x'_a is set to a positive value whenever it is not possible to satisfy the condition which assigns a to a category at least as good as C^h, while a actually belongs to C^h. Impeding the assignment of positive values to x'_a amounts to favor false positive assignments. Hence, positive values of x'_a should be heavily penalized. In contrast, positive values of y'_a correspond to the case in which the conditions for assigning a to the categories above the profile are met while a belongs to the category below the profile. Positive values of y'_a need not be discouraged as much as those of x'_a and therefore we

changed the objective function of the linear program into $\min \sum_{a \in A} 10x'_a + y'_a$.

3. Adjustment of profile. In order to select moves in the profile level on a criterion by a quantity $\pm\delta$, we compute a probability which takes into account the sizes of the sets listed at the end of section 3.1. In all cases, the movements which lower the profile ($-\delta$) are more favorable to false positive than the opposite movements. Therefore, all other things being equal (i.e. the sizes of the sets), the probability of choosing a downward move $-\delta$ should be larger than that of an upward move $+\delta$. The probability of an upward move is thus computed by the following formula

$$P(b_j^h + \delta) = \frac{2|V_{h,j}^{-\delta}| + 1|W_{h,j}^{-\delta} + 0.1|T_{h,j}^{-\delta}|}{|V_{h,j}^{-\delta}| + |W_{h,j}^{-\delta}| + |T_{h,j}^{-\delta}| + 5|Q_{h,j}^{-\delta}| + |R_{h,j}^{-\delta}|}.$$

while that of a downward move is

$$P(b_j^h - \delta) = \frac{4|V_{h,j}^{+\delta}| + 2|W_{h,j}^{+\delta} + 0.1|T_{h,j}^{+\delta}|}{|V_{h,j}^{+\delta}| + |W_{h,j}^{+\delta}| + |T_{h,j}^{+\delta}| + 5|Q_{h,j}^{+\delta}| + |R_{h,j}^{+\delta}|}$$

Remarks The learning algorithm described above is a preliminary version. Many different strategies for learning appropriate concordance and veto profiles as well as their associated weight vectors and thresholds are possible, even with the present option consisting of using successively two variants of the MR-Sort heuristic. In order to assess the latter idea, we report the results of both the first step (concordance part) and the second step (adding the veto) in the experiments done in Section 4. The results of the first part are reported as MR-Sort-FP.

4 Experiments

4.1 Datasets

In view of assessing the performance of the heuristic algorithm designed for learning the parameters of a MR-Sort-CV model, we use it to learn MR-Sort-CV models from several real data sets available at `http://www.uni-marburg.de/fb12/kebi/research/repository/monodata`, which serve as benchmarks to assess monotone classification algorithms [22]. They involve from 120 to 1728 instances, from 4 to 8 monotone attributes and from 2 to 36 categories. In our experiments, categories have been binarized by thresholding at the median. We split the datasets in a twofold 50/50 partition: a learning set and a test set. Models are learned on the first set and evaluated on the test set; this is done 100 times on learning sets drawn at random.

Data set	#instances	#attributes	#categories
DBS	120	8	2
CPU	209	6	4
BCC	286	7	2
MPG	392	7	36
ESL	488	4	9
MMG	961	5	2
ERA	1000	4	4
LEV	1000	4	5
CEV	1728	6	4

Table 1: Data sets

4.2 Results obtained with MR-Sort and NCS

A similar experimental study [20] compares the results obtained with MR-Sort and NCS. The classification accuracy of both methods are provided in Table 2. No significant improvement in classification accuracy was observed when comparing NCS to MR-Sort.

Data set	Heuristic MR-Sort	Heuristic NCS
DBS	0.8377 ± 0.0469	0.8312 ± 0.0502
CPU	0.9325 ± 0.0237	0.9313 ± 0.0272
BCC	0.7250 ± 0.0379	0.7328 ± 0.0345
MPG	0.8219 ± 0.0237	0.8180 ± 0.0247
ESL	0.8996 ± 0.0185	0.8970 ± 0.0173
MMG	0.8268 ± 0.0151	0.8335 ± 0.0138
ERA	0.7944 ± 0.0173	0.7944 ± 0.0156
LEV	0.8408 ± 0.0122	0.8508 ± 0.0188
CEV	0.8516 ± 0.0091	0.8662 ± 0.0095

Table 2: Average and standard deviation of the classification accuracy on the datasets

4.3 Comparing MR-Sort-CV to MR-Sort

In this section, we investigate empirically the benefit obtained by adding "coalitional veto" to MR-Sort, i.e, we compare MR-Sort-CV to MR-Sort. Table 3 provides, for each binarized dataset, the confusion matrices; values provided are mean values of the proportion of alternatives. C^1 and C^2 are the true classes in the dataset, and $\hat{C}^1$ and $\hat{C}^2$ are the computed classifications. The first confusion table contains the proportions obtained with the MR-Sort heuristic. The second confusion table contains the proportions obtained with the MR-Sort-FP heuristic which favors false positives. Finally, the last confusion table contains the proportions obtained with MR-Sort-CV, i.e, when coalitional veto is added to MR-Sort-FP.

These first results show that MR-Sort and MR-Sort-CV provide similar results in terms of classification accuracy and that no benefit is induced from the introduction of coalitional veto. However, it should be noted that MR-Sort-FP obtains only a limited proportion of false positives ($C^2 - \hat{C}^1$). As coalitional veto is able to "correct" alternatives which are overclassified by MR-Sort-FP, we performed a second set of experiments in order to increase the proportion of false positives obtained by MR-Sort-FP (to do so we modify the model selection criterion so that the discarded models are those for which the number of true positive plus 0.9 times the number of false positives plus 0.1 times the number of true negatives is below the median of that quantity on the models' population). The results are provided in Table 4.

These results show that MR-Sort-FP provided higher proportions of false positives, even if this results in a lower overall classification accuracy. MR-Sort-CV is able to significantly improve these results, but the values of classification accuracy for MR-Sort-CV are still, after these changes, similar to the ones of MR-Sort.

These results tend to show that there is no significative improvement in classification accuracy when comparing the results of the standard MR-Sort to the results obtained with MR-Sort-CV. Although MR-Sort-CV is formally a generalization of MR-Sort which brings additional descriptive ability,

Dataset		MR-Sort		MR-Sort-FP		MR-Sort-CV	
		$\hat{C}^1$	$\hat{C}^2$	$\hat{C}^1$	$\hat{C}^2$	$\hat{C}^1$	$\hat{C}^2$
DBS	C^1	42.1	8.5	40.5	10.2	41.8	8.8
	C^2	7.7	41.7	6.0	43.3	7.2	42.2
CPU	C^1	46.7	2.6	46.0	3.2	46.6	2.7
	C^2	4.2	46.6	2.9	47.9	3.3	47.4
BCC	C^1	60.5	10.3	58.1	12.7	63.2	7.6
	C^2	17.2	12.0	17.2	11.9	19.9	9.4
MPG	C^1	44.3	9.2	41.9	11.5	44.5	8.9
	C^2	8.6	37.9	7.9	38.7	9.1	37.4
ESL	C^1	48.6	6.0	46.9	7.6	49.3	5.2
	C^2	4.1	41.4	2.5	42.9	3.8	41.6
MMG	C^1	43.8	7.7	42.4	9.2	44.0	7.5
	C^2	9.6	38.8	8.4	40.0	9.8	38.7
ERA	C^1	69.3	5.1	68.0	6.4	71.8	2.5
	C^2	15.5	10.2	16.9	8.7	18.4	7.3
LEV	C^1	71.1	6.6	69.4	8.4	72.4	5.3
	C^2	9.3	12.9	9.2	13.1	10.9	11.4
CEV	C^1	59.2	10.9	59.2	10.8	61.5	8.5
	C^2	4.1	25.9	4.0	26.0	5.1	24.9

Table 3: Confusion matrices on the datasets for MR-Sort, MR-Sort-FP, and MR-Sort-CV

Dataset		MR-Sort		MR-Sort-FP		MR-Sort-CV	
		$\hat{C}^1$	$\hat{C}^2$	$\hat{C}^1$	$\hat{C}^2$	$\hat{C}^1$	$\hat{C}^2$
DBS	C^1	42.1	8.5	35.4	15.2	39.5	11.2
	C^2	7.7	41.7	2.8	46.6	5.3	44.0
CPU	C^1	46.7	2.6	37.7	11.5	43.9	5.3
	C^2	4.2	46.6	1.4	49.4	4.8	46.0
BCC	C^1	60.5	10.3	6.6	64.3	59.5	11.3
	C^2	17.2	12.0	1.1	28.1	18.8	11.2
MPG	C^1	44.3	9.2	7.7	45.8	44.7	8.8
	C^2	8.6	37.9	0.8	45.8	13.2	33.3
ESL	C^1	48.6	6.0	31.1	23.4	38.7	15.9
	C^2	4.1	41.4	0.7	44.7	3.2	42.3
MMG	C^1	43.8	7.7	8.9	42.6	38.2	13.3
	C^2	9.6	38.8	1.0	47.5	7.8	40.7
ERA	C^1	69.3	5.1	22.2	52.1	56.8	17.5
	C^2	15.5	10.2	3.2	22.5	13.1	12.6
LEV	C^1	71.1	6.6	40.6	37.1	68.2	9.5
	C^2	9.3	12.9	2.3	20.0	13.0	9.3
CEV	C^1	59.2	10.9	56.6	13.5	59.9	10.1
	C^2	4.1	25.9	3.0	27.0	3.4	26.6

Table 4: Confusion matrices for MR-Sort, MR-SortFP, and MR-Sort-CV when strengthening the bias in favor of false positives

the experiments fail to show an improvement on the ability of MR-Sort-CV to classify the benchmark datasets better than MR-Sort.

These results seem to us preliminary as it is not straightforward to state whether the inability to improve the result by the addition of coalitional veto comes from an insufficient performance of the algorithm, or from the limited additional descriptive ability induced by the introduction of coalitional veto to MR-Sort. Further analysis should be conducted.

5 Conclusion

We have presented MR-Sort-CV a new original extension of the MR-Sort ordered classification model. This model introduces a new and more general form of veto condition which applies on coalitions of criteria rather than a single criterion. This coalitional veto condition can be expressed as a reversed MR-Sort rule. Such a symmetry enables us to design a heuristic model to learn an MR-Sort-CV model, based on the use of an algorithm to learn MR-Sort. Preliminary results are interesting, but further investigations are needed to take benefit of this new ordered classification model.

REFERENCES

[1] Bouyssou, D., Marchant, T.: An axiomatic approach to noncompensatory sorting methods in MCDM, I: The case of two categories. European Journal of Operational Research 178(1), 217–245 (2007)

[2] Bouyssou, D., Marchant, T.: An axiomatic approach to noncompensatory sorting methods in MCDM, II: More than two categories. European Journal of Operational Research 178(1), 246–276 (2007)

[3] Bouyssou, D., Pirlot, M.: A characterization of concordance relations. European Journal of Operational Research 167(2), 427–443 (2005)

[4] Bouyssou, D., Pirlot, M.: Further results on concordance relations. European Journal of Operational Research 181, 505–514 (2007)

[5] Cailloux, O., Meyer, P., Mousseau, V.: Eliciting ELECTRE TRI category limits for a group of decision makers. European Journal of Operational Research 223(1), 133–140 (2012)

[6] Chateauneuf, A., Jaffray, J.Y.: Derivation of some results on monotone capacities by Möbius inversion. In: Bouchon-Meunier, B., Yager, R.R. (eds.) Uncertainty in Knowledge-Based Systems, International Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems, IPMU '86, Paris, France, June 30 - July 4, 1986, Selected and Extended Contributions. Lecture Notes in Computer Science, vol. 286, pp. 95–102. Springer (1986), http://dx.doi.org/10.1007/3-540-18579-8_8

[7] Dias, L., Mousseau, V., Figueira, J.R., Clímaco, J.: An aggregation/disaggregation approach to obtain robust conclusions with ELECTRE TRI. European Journal of Operational Research 138(1), 332–348 (2002)

[8] Doumpos, M., Marinakis, Y., Marinaki, M., Zopounidis, C.: An evolutionary approach to construction of outranking models for multicriteria classification: The case of the ELECTRE TRI method. European Journal of Operational Research 199(2), 496–505 (2009)

[9] Doumpos, M., Zopounidis, C.: Multicriteria Decision Aid Classification Methods. Kluwer Academic Publishers (2002)

[10] Leroy, A., Mousseau, V., Pirlot, M.: Learning the parameters of a multiple criteria sorting method. In: Brafman, R., Roberts, F., Tsoukiàs, A. (eds.) Algorithmic Decision Theory, Lecture Notes in Artificial Intelligence, vol. 6992, pp. 219–233. Springer (2011)

[11] Mousseau, V., Figueira, J.R., Naux, J.P.: Using assignment examples to infer weights for ELECTRE TRI method: Some experimental results. European Journal of Operational Research 130(1), 263–275 (2001)
[12] Mousseau, V., Słowiński, R.: Inferring an ELECTRE TRI model from assignment examples. Journal of Global Optimization 12(1), 157–174 (1998)
[13] Ngo The, A., Mousseau, V.: Using assignment examples to infer category limits for the ELECTRE TRI method. Journal of Multi-criteria Decision Analysis 11(1), 29–43 (2002)
[14] Olteanu, A.L., Meyer, P.: Inferring the parameters of a majority rule sorting model with vetoes on large datasets. In: DA2PL 2014 : From Multicriteria Decision Aid to Preference Learning. pp. 87–94 (2014)
[15] Roy, B.: The outranking approach and the foundations of ELECTRE methods. Theory and Decision 31, 49–73 (1991)
[16] Roy, B., Bouyssou, D.: Aide multicritère à la décision: méthodes et cas. Economica Paris (1993)
[17] Sobrie, O., Mousseau, V., Pirlot, M.: Learning a majority rule model from large sets of assignment examples. In: Perny, P., Pirlot, M., Tsoukiás, A. (eds.) Algorithmic Decision Theory. Lecture Notes in Artificial Intelligence, vol. 8176, pp. 336–350. Springer, Brussels, Belgium (2013)
[18] Sobrie, O., Mousseau, V., Pirlot, M.: Learning the parameters of a multiple criteria sorting method from large sets of assignment examples. In: 77th meeting of the EWG on MCDA. Rouen, France (April 2013)
[19] Sobrie, O., Mousseau, V., Pirlot, M.: Learning the parameters of a majority rule sorting model taking attribute interactions into account. In: DA2PL 2014 Workshop From Multiple Criteria Decision Aid to Preference Learning. pp. 22–30 (2014), http://www.lgi.ecp.fr/~mousseau/DA2PL-2014, paris, France
[20] Sobrie, O., Mousseau, V., Pirlot, M.: Learning the parameters of a non compensatory sorting model. In: Walsh, T. (ed.) Algorithmic Decision Theory. Lecture Notes in Artificial Intelligence, vol. 9346, pp. 153–170. Springer, Lexington, KY, USA (2015)
[21] Sobrie, O., Pirlot, M., Mousseau, V.: New veto relations for sorting models. Tech. rep., Laboratoire Génie Industriel, Ecole Centrale Paris (October 2014), http://www.lgi.ecp.fr/Biblio/PDF/CR-LGI-2014-04.pdf, cahiers de recherche 2014-04
[22] Tehrani, A.F., Cheng, W., Dembczyński, K., Hüllermeier, E.: Learning monotone nonlinear models using the Choquet integral. Machine Learning 89(1–2), 183–211 (2012)
[23] Yu, W.: Aide multicritère à la décision dans le cadre de la problématique du tri: méthodes et applications. Ph.D. thesis, LAMSADE, Université Paris Dauphine, Paris (1992)
[24] Zheng, J., Metchebon, S.A., Mousseau, V., Pirlot, M.: Learning criteria weights of an optimistic Electre Tri sorting rule. Computers & OR 49(0), 28–40 (2014), http://www.sciencedirect.com/science/article/pii/S0305054814000677
[25] Zopounidis, C., Doumpos, M.: Multicriteria classification and sorting methods: A literature review. European Journal of Operational Research 138(2), 229–246 (April 2002)

Incorporating Auxiliary Data into Recommender Systems

Thomas Spura[1], Michael Baumann[1] and Artus Krohn-Grimberghe[1]

Abstract. Recommender Systems predict the ratings that a user would give to unseen/unbought items. However, the huge sparsity of user $\times$ item ratings makes this task challenging. In this paper, we incorporate auxiliary data describing the inner connection of the items to enhance the quality of the recommender system's predictions and present an approach to improve the dimensionality of the data.

1 Introduction

Recommender systems are widely used in online services to provide useful and relevant *items* to *users*. In this paper we deal with movie recommendations and thus, we focus on movies or videos as items. Popular examples for recommender systems are YouTube and Netflix, where new movies are recommended based on the previous searches and watching histories of the users. Improvements in such recommender systems help increasing the user experience. In this paper, we propose a new algorithm to incorporate auxiliary data of those movies—such as information about the genre and the cast— in order to recommend more relevant items.

Recommender systems are broadly classified into three categories [1] which are content based, collaborative filtering [7] and hybrid approaches [3]. Content based approaches [9] build profiles for users and movies on which predictions are based. Collaborative filtering approaches [4] match users with a similar watching history and predict movies that a particular user might like based on these groups of "similar" users. Hybrid models try to connect and unify both approaches mentioned above. Other models like tensor factorization that improve a recommender system with contextual data have also been proposed [6, 15, 8].

In general, recommender systems deal with a set of users (U) and a set of items (I). These users have rated a subset of those items (here movies). The rating information is stored in a user, items rating matrix which has U rows and I columns. Besides this, we have additional information in the form of auxiliary data about the items. Here, a short summary as well as information about the cast (e.g. actors, producers, directors, etc.) of the movie is used. This information needs to be in a form such that it can be linked to the corresponding items. For example, we can directly derive mappings movie $\rightarrow$ actor, movie $\rightarrow$ genre, movie $\rightarrow$ producers, etc. from the data. Together with the user $\times$ movie ratings, the auxiliary information is then "joined" using the movie id to produce an higher-dimensional interaction matrix that is then process by a Factorization Machine (FM) [10].

[1] University of Paderborn, Chair of Analytic Information Systems and Business Intelligence, email: {tomspur,baumann,artus}@aisbi.de

2 Addition of Auxiliary Data

The new approach is to model the latent factors in the FMs with auxiliary data which is explained in the following.

2.1 Genre and Cast Auxiliary Data

The genre information is parsed from the abstract of the given movie and afterwards, each word is stemmed to their root form. With this movie information, a movie $\times$ genre (MG) matrix is obtained which is then normalized. This matrix MG models the connection of movie to genres and also considers the fact that movies might belong to more than on genre. The user $\times$ movie matrix is now jointly factorized together with the latent factor of genre to obtain a user $\times$ genre matrix using the approach of Singh and Gordon [13].

The information about the cast of the movie is incorporated analogous to the approach for the genre information to obtain a user $\times$ actor matrix.

2.2 Dimensionality Reduction and Clustering

Due to the huge number of members in the cast (up to 1982 people), the computational complexity to estimate new recommendations is very high.

The first approach to reduce the computational complexity of above algorithm is to reduce the maximum cast count. Thus, the dimensionality is reduced by considering only the n_{cast} most occurring persons among all movies. In initial experiments $n_{cast} = 100$ was a reasonable value.

To keep as much information of the cast as possible but still reduce the dimensionality of the user $\times$ cast matrices, the casts of different movies can be condensed with a clustering approach. This is achieved by grouping similar main characters, directors, producers, etc. each using cosine similarity between different persons.

3 Factorization Machines

In our opinion, a well suited option to integrate—esp. very sparse—side information into factorization schemes are Factorization Machines [10], which have been specifically designed to cope with huge sparse matrices and can be compared to SVMs with a polynomial kernel, that attempt to model further higher order interactions among the variables and does not assume them as independent variables. The model equation of an FM for pairwise interactions is defined as [10]

$$\hat{y}(\mathbf{x}) = w_0 + \sum_{i=1}^{n} w_i x_i + \sum_{i=1}^{n} \sum_{j=i+1}^{n} \langle \mathbf{v}_i, \mathbf{v}_j \rangle \, x_i x_j \,, \qquad (1)$$

where $\hat{y}$ is the target variable (rating), $\mathbf{x}$ is the feature vector (consisting of the rated movies and the additional metadata), $w_0 \in \mathbb{R}$

describes the global bias, $w_i \in \mathbb{R}$ describes the strength of the i-th variable and $\langle v_i, v_j \rangle$ describes the interaction of the i-th and j-th variable. A row v_i within V describes the i-th variable with k factors, $k \in \mathbb{N}$ being a hyper-parameter that defines the dimensionality of the factorization

$$\langle v_i, v_j \rangle = \sum_{f=1}^{k} v_{i,f} v_{j,f} \; . \tag{2}$$

This equation helps at modelling related interactions in the feature vector, even in sparse conditions [10].

The auxiliary data that is now added to the feature vector **x** from Equation 1 consisting of features from the genres and additional cast information of the movies. Whereas the latter was directly obtained from DBpedia [2], the former is generated from the abstract of the movies. This is achieved by extracting keywords from the abstract and stemming them to their root form. Furthermore, the genres have been normalized per movie by the total number of genres one movie contained to account for multiple genres per movie.

The additional cast information is huge and compuational demanding. To reduce the computational cost and avoid information loss similar actors, directors, producers etc. are combined. This is done by calculating a cosine similarity matrix between each cast groups and the genres they played. Using clustering on this cosine similarity matrix, a reduced matrix is obtained and similar actors, directors, producers etc. are grouped and added to the final feature vector.

4 Experimental Results

This approach is tested on the MovieLens 1M dataset [5], whereas the additional auxiliary data for each movie derived from DBpedia [2] consists of a summary of each movie and information about its cast. The synopsis was used to add a tag to the movie based on genre, nominations it received, etc. The direct genre information from the MovieLens dataset was not used in order to generalize the process of obtaining relevant tags directly from the synopsis. The data is evaluated via five-fold cross validation and the hyper-parameters were tuned on the validation data, which was taken to be 10% of the total data. FMs were used as implemented in libFM [11] and different converging algorithms to obtain the interactions are compared (Stochastic Gradient Descent (SGD), Alternating Least Square (ALS), and Markov Chain Monte Carlo (MCMC)).

The results are shown in Table 1 for the different approaches through which factorization machines can be implemented and with all the different auxiliary data that can be added to them. The SGD approach is of equal performance to the matrix factorization from our initial testing and the RMSE values are very close. (0.8598 for the raw data earlier and 0.8571 when implemented through factorization machine). Out of the different converging algorithms that are implemented in libFM, MCMC performs the best, followed by SGD and then by ALS. In terms of percentage change, the most change is recorded in MCMC, followed by ALS and then SGD.

	Metadata			
Method	none (*)	Cast	Genre	Cast & Genre
ALS	0.8681	0.8653	0.8648	**0.8628**
SGD	0.8571	0.8567	0.8539	**0.8526**
MCMC	0.8419	0.8404	0.8375	**0.8357**

Table 1. RMSE values for our approach compared to baseline models (*).

5 Conclusion

In this paper, a new approach was proposed to incorporate auxiliary data to recommender systems in order to broaden the feature space and improve the performance of factorization machines. The predictions were superior to common matrix factorizations as well as to Factorization Machines without auxiliary data. As future work, we like to test this approach on a broader range of data sets and further explore the generation of relevant tags such as e.g. in [14, 12] or employing other means of natural language processing. Additionally, the importance of auxiliary data per se as well as different prepocessing steps will be explored.

REFERENCES

[1] Gediminas Adomavicius and Alexander Tuzhilin, 'Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions', *IEEE Trans. Knowl. Data Eng.*, **17**(6), 734–749, (2005).

[2] Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary G. Ives, 'Dbpedia: A nucleus for a web of open data', in *The Semantic Web, 6th International Semantic Web Conference, 2nd Asian Semantic Web Conference, ISWC 2007 + ASWC 2007*, Lecture Notes in Computer Science, pp. 722–735. Springer, (2007).

[3] Li Chen, Guanliang Chen, and Feng Wang, 'Recommender systems based on user reviews: the state of the art', *User Model. User-Adapt. Interact.*, **25**(2), 99–154, (2015).

[4] Michael D. Ekstrand, John Riedl, and Joseph A. Konstan, 'Collaborative filtering recommender systems', *Foundations and Trends in Human-Computer Interaction*, **4**(2), 175–243, (2011).

[5] F. Maxwell Harper and Joseph A. Konstan, 'The movielens datasets: History and context', *TiiS*, **5**(4), 19, (2016).

[6] Alexandros Karatzoglou, Xavier Amatriain, Linas Baltrunas, and Nuria Oliver, 'Multiverse recommendation: n-dimensional tensor factorization for context-aware collaborative filtering', in *Proceedings of the 2010 ACM Conference on Recommender Systems, RecSys 2010*, pp. 79–86. ACM, (2010).

[7] Yehuda Koren, 'Collaborative filtering with temporal dynamics', *Commun. ACM*, **53**(4), 89–97, (2010).

[8] Volodymyr Kuleshov, Arun Tejasvi Chaganty, and Percy Liang, 'Tensor factorization via matrix factorization', in *Proceedings of the Eighteenth International Conference on Artificial Intelligence and Statistics, AISTATS 2015*, volume 38 of *JMLR Workshop and Conference Proceedings*. JMLR, (2015).

[9] Michael J. Pazzani and Daniel Billsus, 'Content-based recommendation systems', in *The Adaptive Web, Methods and Strategies of Web Personalization*, volume 4321 of *Lecture Notes in Computer Science*, pp. 325–341. Springer, (2007).

[10] Steffen Rendle, 'Factorization machines', in *ICDM 2010, The 10th IEEE International Conference on Data Mining*, pp. 995–1000. IEEE Computer Society, (2010).

[11] Steffen Rendle, 'Factorization machines with libFM', *ACM Trans. Intell. Syst. Technol.*, **3**(3), 57:1–57:22, (2012).

[12] Giovanni Semeraro, Pasquale Lops, Pierpaolo Basile, and Marco de Gemmis, 'Knowledge infusion into content-based recommender systems', in *Proceedings of the 2009 ACM Conference on Recommender Systems, RecSys 2009*, pp. 301–304. ACM, (2009).

[13] Ajit Paul Singh and Geoffrey J. Gordon, 'Relational learning via collective matrix factorization', in *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 650–658. ACM, (2008).

[14] Márcio Soares and Paula Viana, 'Tuning metadata for better movie content-based recommendation systems', *Multimedia Tools Appl.*, **74**(17), 7015–7036, (2015).

[15] Hendrik Wermser, Achim Rettinger, and Volker Tresp, 'Modeling and learning context-aware recommendation scenarios using tensor decomposition', in *International Conference on Advances in Social Networks Analysis and Mining, ASONAM 2011*, pp. 137–144. IEEE Computer Society, (2011).

Semi-Parametric Estimation for Paired Comparisons Using SDP

Ivo Fagundes David de Oliveira[1] and **Nir Ailon**[2] and **Ori Davidov**[3]

Abstract. We assume linear stochastic transitivity and address the problem of estimating the underlying probability matrix, the merit of the objects being compared and the underlying function from a paired comparison experiment. Our formulation yields a semi-definite programming problem that we use as a refinement step for a given estimator of the paired comparison probability matrix. We provide a detailed sensitivity analysis and as a result we extract statistical properties of the resulting estimator. By building on previous results and our sensitivity analysis we also provide bounds on the expected squared error of the estimated probability matrix within the round-robin setting and a paired comparison experiment with random encounters. Our novel contribution recovers not only the merits of the players within the game (which is the classical paired comparison setting) but also the underlying structure of the game described by the paired comparison function. Our methodology is illustrated with numerical experiments.

1 Introduction

A paired comparison ranking method is a procedure to infer a preference relation from the observation of a finite amount of pairwise comparisons, typically with binary outcomes. Ranking systems have been used in many different contexts; examples of applied ranking theory are commonly found in the following literature: sports and games, to rank and pair contestants [7]; psychometrics, to model and predict preference relations [11]; machine learning, to infer generative models [12]; statistics, to estimate underlying probabilities [6].

The Linear Stochastic Transitivity Model Within the framework of statical ranking models, linear stochastic transitivity models are arguably amongst the most attractive options due to their simplicity. Linear stochastic transitivity (LST) assumes that: for each player i (also called object, item or category to be ranked etc.) there exists a merit $\mu_i \in \mathbb{R}$ (also called skill, score or rating etc.) and there exists a distribution function $F : \mathbb{R} \to [0,1]$ increasing and with $F(x) + F(-x) = 1$ such that for each pair of players (i,j) the probability of player i defeating j (or being preferred over etc.) is given by $p_{ij} = F(\mu_i - \mu_j)$. We will adopt the gaming notation from hereon, that is, players are compared in games and we are interested in finding their underlying merits.

Two canonical linear ranking models widely studied and used in literature are the Thurstonian model [14], hereby called the THU model, and the (Zarmelo) Bradley-Terry-Luce model, hereby called the BTL model [3]. These canonical approaches inasmuch as they have served as cornerstones in ranking literature require rigid conditions on the underlying model in order to yield the sound statistical interpretations that they offer, the strongest of which is the assumption that function F is known. In our work we will weaken the hypothesis that F is known and assume only that it belongs to a parametric family of functions; we will then estimate not only the underlying probability but also the underlying linear stochastic function.

Background and Notation Given a finite amount I of players labeled from $i = 1, ..., I$, let W_{ijk} be independent binary random variables where $W_{ijk} = 1$ if player i wins against player j on their k^{th} encounter and $W_{ijk} = 0$ otherwise. We assume each encounter yields a binary outcome, therefore $W_{ijk} + W_{jik} = 1$. We assume a finite amount of encounters was observed between each pair of players, namely m_{ij}. We define $W_{ij} = \sum_{k=1}^{m_{ij}} W_{ijk}$ to be the number of games player i won against player j in these m_{ij} encounters. In our model, this will be treated as a given, and though questions related to how to pair players be compellingly interesting it is beyond the scope of this article. We assume there exists an underlying probability matrix $\boldsymbol{P} = [p_{ij}]_{ij=1,...,I} \in [0,1]^{I \times I}$ where p_{ij} is the probability that player i defeats player j in an encounter, that is, $\mathbb{P}(W_{ijk} = 1) = p_{ij}$. We refer to vectors in boldface and matrices in capital letters and boldface, that is, $\mathbf{W} \equiv [W_{ij}]_{i,j=1,...,I}$, $\boldsymbol{\mu} = [\mu_i]_{i=1,...,I}$, $\mathbf{P} = [p_{ij}]_{i,j=1,...,I}$ and $\mathbf{M} = [m_{ij}]_{i,j=1,...,I}$.

One standard ranking problem is to estimate $\boldsymbol{P}$ and/or $\boldsymbol{\mu}$ given the (summary) observable $\boldsymbol{W}$ and a pre-specified function F. A natural choice for performing such estimation is via least squares estimation (LSE), for often, least squares yield simple and easily computable procedures. Many least-squares ranking procedures have been proposed, typically the least squares methods have as input estimates $\hat{\Delta}_{ij}$ of the difference between the merit μ_i and μ_j. Under the linear stochastic transitivity assumption, a natural way of obtaining $\hat{\Delta}_{ij}$ is by defining $\hat{\Delta}_{ij} \equiv F^{-1}(\hat{p}_{ij})$ where $\hat{p}_{ij}$ is typically the empirical frequency $\hat{f}_{ij} = \frac{W_{ij}}{W_{ij}+W_{ji}}$, or Laplace's estimator $\tilde{p}_{ij} = \frac{W_{ij}+1}{W_{ij}+W_{ji}+2}$ which provides more stable estimators for low sample sizes. The procedure is then to calculate $\hat{\boldsymbol{\mu}}$ as a least-squares estimate defined by the following optimization problem:

$$\hat{\boldsymbol{\mu}} \in \operatorname{argmin} \sum_{i \neq j} \Omega_{ij} \left(\hat{\Delta}_{ij} - (\mu_i - \mu_j) \right)^2 \tag{1}$$

Where Ω_{ij} are given weights; typically proportional to the variance of the estimated $\hat{\Delta}_{ij}$, or proportional to the number of games observed between player i and j, that is $\Omega_{ij} = m_{ij}$. If we assume the weights Ω_{ij} are symmetric, that is $\Omega_{ij} = \Omega_{ji}$, and also $\Omega_{ii} = -\sum_{j:j \neq i} \Omega_{ij}$ and $\Delta_{ii} = 0$, then $\boldsymbol{\mu}$ is a solution to (1) if and only if it satisfies the linear equation:

[1] Computer Science Department, Technion, email: ivodavid@gmail.com
[2] Computer Science Department, Technion, email: nailon@cs.technion.ac.il
[3] Statistics Department, Haifa University, email: davidov@stat.haifa.ac.il

$$\boldsymbol{\Omega} \cdot \boldsymbol{\mu} = -\left[\boldsymbol{\Omega} \bullet \hat{\boldsymbol{\Delta}}\right] \cdot \mathbf{1} \tag{2}$$

Where $[\boldsymbol{\Omega} \bullet \hat{\boldsymbol{\Delta}}]$ is the matrix formed by the element-wise multiplication $[\boldsymbol{\Omega} \bullet \hat{\boldsymbol{\Delta}}] = [\Omega_{ij}\hat{\Delta}_{ij}]_{ij=1,\dots,I}$. The linear system (2) typically admits multiple solutions, and to eliminate the degrees of freedom, one may adopt additional constraints such as $\sum_i \mu_i = 0$. Furthermore, if the weights are all unitary, then the closed form solution to (2) is known as the row-sum procedure, that is $\hat{\mu}_{ij} = \frac{1}{I}\sum_j \Delta_{ij}$.

Many versions of least squares approaches can be found in literature and their properties have been extensively studied. We refer the reader to three references for more properties of least squares rankings under different names, refer to "Hodge rank" in [9], "least squares" in [5] and "row-sum procedure" in [8].

Linear models mainly differ in their definition of function F. The THU model and the BTL model are derived respectively from the normal and the logistic distribution function. Other less popular linear models are also studied in literature, we mention the Threshold model (THR) derived from models of animal behavior [16] which gives yield to a Laplacian distribution and also the locally linear model (LIN) which is found in theoretical results such as in [1] which corresponds to a uniform distribution.

These traditional approaches to estimation, that is, least-squares and maximum-likelihood methods [15], depend strongly on one common assumption, namely that function F is known. Dropping the assumption that F is known gives yield to non-trivial mathematical problems. The very characterization of the set of matrices $\boldsymbol{P} \in \mathbb{R}^{k\times k}$ that are consistent with the LST assumption has been an open question in literature since 1959 [2]. Recent interest has been rekindled in performing estimation of the probability matrix $\boldsymbol{P} \in [0,1]^{I\times I}$ under sub-LST models, namely the Strong Stochastic Transitivity model (from now on to be called SST) which can be seen as an exterior approximation to the LST condition (for more details refer to [4, 13] and the references therein). The recent works of [4, 13], for example, have characterized minimax bounds on the expected squared error of estimators under both LST and SST conditions; we will make use of these recent results to show that our estimation procedure also achieves optimal mini-max rates under the proposed setting.

Our Contribution Previous methods have succeeded in estimating the underlying paired comparison probability matrix, we are unaware though of any successful attempt in recovering function F through a paired comparison experiment. Our main contribution takes as input an estimate of the probability matrix and returns an estimate of the underlying function F, the merit vector and a refinement the original estimate of the probability matrix. Our procedure considers a parametric family of functions and the refined probability matrix can be understood as an interior approximation (with arbitrary precision) to the LST hypothesis giving yield to a semi-definite programming problem with tractable solution. Our second contribution is a thorough sensitivity analysis of our problem formulation by which we derive statistical properties such as consistency and minimax expected squared error bounds on the refined estimator.

2 Problem Formulation

Least Squares Estimation Over Polynomial Families Before we describe our main problem, we will revisit problem (1) in order to make some necessary remarks concerning the notation and the generalization of the least squares estimator.

Given a strictly increasing continuous distribution function F we define $F^{-1}(\boldsymbol{P})$ acting on each element individually $F^{-1}(\boldsymbol{P}) = [F^{-1}(p_{ij})]_{ij=1,\dots,I}$. We will also define the linear operator Δ from $\mathbb{R}^k \to \mathbb{R}^{k\times k}$ such that $\Delta\boldsymbol{\mu} \equiv [\mu_i - \mu_j]_{ij=1,\dots,k}$. This way problem (1) with $\hat{\Delta}_{ij} = F^{-1}(\hat{p}_{ij})$ can be explicitly written as:

$$\hat{\boldsymbol{\mu}} \in \text{argmin}_{\boldsymbol{\mu}} ||F^{-1}(\hat{\boldsymbol{P}}) - \Delta\boldsymbol{\mu}||$$

Where the norm is the (weighted or unweighted) L_2 norm; notice that in general we can choose any convenient norm such as the L_1 or L_∞ norms (which give yield to tractable linear programs). Notice that the least squares procedure takes an estimator $\hat{\boldsymbol{P}}$ of $\underline{\boldsymbol{P}}$ and therefore $\hat{\boldsymbol{P}}^* = F(\Delta\hat{\boldsymbol{\mu}})$ can be seen as a refinement of $\hat{\boldsymbol{P}}$, an estimator of $\underline{\boldsymbol{P}}$. The refined estimator $\hat{\boldsymbol{P}}^* = F(\Delta\hat{\boldsymbol{\mu}})$ is the result of a projection of $F^{-1}(\hat{\boldsymbol{P}})$ to the image set of the operator Δ making it thus a feasible probability matrix generated by function F and vector $\boldsymbol{\mu}$. We wish, though, to address the case when function F is not known.

In our setting in order to weaken the assumption that function F is known, we will assume that function F belongs to the following parametric set:

$$\boldsymbol{\mathcal{F}} \equiv \{F_c \mid \exists \boldsymbol{c} \in \mathbb{R}^{J+1} \text{ st. } F_c \text{ is increasing, } F_c(x) + F_c(-x) = 1,$$

$$F_c^{-1}(p) = c_0 + c_1 p + \dots + c_J p^J \text{ for } p \in [0,1/2] \text{ with } ||\boldsymbol{c}||_1 \le U\}$$

Where J and U are given constants. That is, it will be assumed that the inverse of the distribution function F_c is a polynomial over the interval $[0,1/2]$, and therefore for $p \in [1/2,1]$ due to the relation $F_c(x) + F_c(-x) = 1$ we have $F_c^{-1}(p) = -F_c^{-1}(1-p)$ which is also a polynomial. This way, given $\hat{\boldsymbol{P}} \in \mathbb{R}^{I\times I}$ an estimator of matrix $\underline{\boldsymbol{P}} \in \mathbb{R}^{I\times I}$, $J \in \mathbb{N}$ and $U < \infty$ a natural way to generalize problem (1) is:

$$(\hat{\boldsymbol{\mu}} \;\; \hat{\boldsymbol{c}}) \in \text{argmin}_{\boldsymbol{\mu}, F_c \in \boldsymbol{\mathcal{F}}} ||F_c^{-1}(\hat{\boldsymbol{P}}) - \Delta\boldsymbol{\mu}|| \tag{3}$$

This way, in addition to refining the estimate $\hat{\boldsymbol{P}}$ and estimating $\boldsymbol{\mu}$, we can also recover function F simultaneously. Our main results apply to the L_1, L_2 and L_∞ norms, though, for simplicity, we will display throughout the main bulk of the article that which concerns the L_2 norm. We note also that the weighted version of the refinement procedure is analogously defined. An extended version of the article that includes the proofs of our results and analogous results for the L_1 and L_∞ norms will be provided by the author upon request.

The set $\boldsymbol{\mathcal{F}}$ though much smaller than the set of all distribution functions, can arbitrarily approximate any continuous increasing function for sufficiently large J and U. In fact for a fixed amount of players I the value of function F is only relevant on less than I^2 points, and so function F must be well approximated only on these points. Our parametric set should be, therefore, expected to be sufficiently appropriate for $J < O(I^2)$ and sufficiently large U.

Our main concerns now are to characterize the constraints on the coefficients $\boldsymbol{c}$ of the inverse of F, given the constraints on F . Due to the constraint $F(x) + F(-x) = 1$ and the definition of the set $\boldsymbol{\mathcal{F}}$, we need only characterize the coefficients $\boldsymbol{c}$ of $F_{\boldsymbol{c}}^{-1}(p) = c_0 + c_1 p + \dots + c_J p^J$ for $p \in [0,1/2]$ and then for $p \in [1/2,1]$ function $F_{\boldsymbol{c}}^{-1}$ is uniquely determined by $F^{-1}(p) = -F^{-1}(1-p)$. The constraints on the distribution function F over interval $[0,1/2]$ are: (a) Function F is an increasing function (b) Function F evaluated at 0 has value $F(0) = 1/2$.

We will show how to characterize these two constraints in terms of convex constraints. Our characterization result stems from a a description of (a) as a semi-definite constraint on the parameter space attributed to the Polya-Szego, Fekete, and Markov-Lukacs [10] and a description of (b) as a set of linear constraints on the parameter space.

Definition 1. *Let $P(x)$ be a polynomial function, then, $P(x)$ is said to be a sum of squares of polynomials if $P(x) = \sum_i (P_i(x))^2$ where for each i $P_i(x)$ is a polynomial.*

We use the following two results that can be found in [10]:

Lemma 1 (Sum of Squares Polynomials)**.** *Let $P(x)$ be a polynomial of degree at most $2d$. Then the following conditions are equivalent:*

1. *$P(x) = \sum_{i=0}^{2d} p_i x^i$ is a sum of squares of polynomials;*
2. *There exists $\boldsymbol{Q} \in \mathbb{S}_+^{d+1}$ such that $p_i = \sum_{j+k=i} Q_{jk}$;*

Where $\mathbb{S}_+^{d+1} \subset \mathbb{R}^{d+1 \times d+1}$ is the set of symmetric PSD matrices.

Theorem 1 (Non-negative Polynomials)**.** *Let $P(x)$ be a polynomial of even degree. Then $P(x)$ is non-negative over the interval $[a, b]$ if and only if it can be written as: $P(x) = s(x) + (x-a)(b-x)t(x)$ where $s(x)$ and $t(x)$ are sum of squares polynomials of degree at most $2d$ and $2d-2$ respectively.*

Our result is then the characterization of the parameters $\boldsymbol{c} \in \mathbb{R}^{J+1}$ of the inverse function $F_{\boldsymbol{c}}^{-1}$ that satisfy (a) and (b), an outline of the proof is also provided.

Theorem 2 (Set Characterization)**.** *Given $J = 2d+1 \in \mathbb{N}$ an odd number and $U < \infty$. Let k be any natural number, define $\mathbb{S}_+^k$ as the set of symmetric PSD matrices in $\mathbb{R}^{k \times k}$. Then $F \in \boldsymbol{\mathcal{F}}$ if and only if there exists $\boldsymbol{Q^0} \in \mathbb{S}_+^{d+1}$ and $\boldsymbol{Q^1} \in \mathbb{S}_+^{d}$ such that[4]:*

$$\begin{cases} s_i = \sum_{j+k=i} Q^0_{jk} & \text{for } i = 0, ..., J-1 \\ t_i = \sum_{j+k=i} Q^1_{jk} & \text{for } i = 0, ..., J-3 \\ (i+1) \cdot c_{i+1} = \frac{1}{2} t_{i-1} - t_{i-2} + s_i & \text{for } i = 0, ..., J-1 \\ \sum_{i=0}^{J} c_i \left(\frac{1}{2}\right)^i = 0 & ||\boldsymbol{c}||_1 \leq U \end{cases} \tag{4}$$

where t_{J-1}, t_{J-2},t_{-1} and t_{-2} are defined to be null.

Proof (Sketch): Function F is increasing if and only if F^{-1} is also increasing, therefore constraint (a) is equivalent to monotonicity of F^{-1}. By the definition of F we have that $F^{-1}(p) = -F^{-1}(1-p)$, and therefore F^{-1} is monotonic over the interval $[0,1]$ if and only if it is monotonic over the interval $[0, 1/2]$. Since $F^{-1}(p) = \sum_{i=0}^{J} c_i p^i$ for $p \in [0, 1/2]$, F^{-1} is monotonic if and only if $\frac{d}{dp} F^{-1}(p) = c_1 + 2 \cdot c_2 p + ... + J \cdot c_J p^{J-1}$ is non-negative over the interval $[0, 1/2]$. Combining this with Theorem 1 and Lemma 1 we obtain the first three equations.

From constraint (b) we have that $F(0) = 1/2$, this is equivalent to $F^{-1}(1/2) = 0$ which is equivalent to $\sum_{i=0}^{J} c_i \left(\frac{1}{2}\right)^i = 0$, this and the fact that $||\boldsymbol{c}||_1 \leq 0$ provide the last two constraints of (4). □

Corollary 1. *Given that $\hat{p}_{ij} + \hat{p}_{ji} = 1$ then, problem (3) for the L_2 norm is equivalent to the semi-definite programming problem:*

$$\begin{cases} \min \sum \left(c_0 + c_1 \hat{p}_{ij} + ... + c_J \hat{p}_{ij}^J + \mu_j - \mu_i\right)^2 \\ \quad i \neq j, \hat{p}_{ij} < \frac{1}{2} \\ \quad i < j, \hat{p}_{ij} = \frac{1}{2} \qquad \text{subject to (4)} \end{cases} \tag{5}$$

Similar to problem (1), additional constraints should be added to avoid trivial solutions. We include the equality constraint $\sum \mu_i = 0$ and the equality constraint $F'(0) = 1$ to avoid degenerate solutions. These constraints are also convex, in fact, constraints of the form $F_{\boldsymbol{c}}(\Delta\boldsymbol{\mu}) = \boldsymbol{p_0}$ and $F'_{\boldsymbol{c}}(\Delta\boldsymbol{\mu}) = \boldsymbol{p_0}'$ are linear. Further on, in Section 6, we discuss guidelines on how to chose additional constraints to avoid degenerate solutions and underdeterminacy.

This formulation is an extension to the least squares estimate over linear stochastic transitivity functions and is a tractable convex optimization problem. Notice that analogous corollaries can be stated for the L_1 and L_∞ norms. We will analyze the following refinement step, from now on referred to as the Semi-Parametric Refinement (SPR):

Algorithm: Semi-Parametric Refinement

1. *Input $(\hat{\boldsymbol{P}}, J, ||\cdot||, U)$. Where $\hat{\boldsymbol{P}} \in \mathbb{R}^{I \times I}$ is an estimate of the true underlying probability matrix $\underline{\boldsymbol{P}} \in \mathbb{R}^{I \times I}$ with $\hat{p}_{ij} + \hat{p}_{ji} = 1$; $J \in \mathbb{N}$, an odd number, is the degree of the polynomial function $F_{\boldsymbol{c}}^{-1}(p) = c_0 + c_1 p + ... + c_J p^J$ for $p \in [0, 1/2]$, $||\cdot||$ is the L_1, L_2 or L_∞ norm and U is an upper-bound on the L_1 norm of $\boldsymbol{c}$.*
2. *Return $(\hat{\boldsymbol{\mu}}\ \hat{\boldsymbol{c}}) \in \begin{cases} \text{argmin} & ||F_{\boldsymbol{c}}^{-1}(\hat{\boldsymbol{P}}) - \Delta\boldsymbol{\mu}|| \\ \text{st.} & \boldsymbol{\mu} \in \mathbb{R}^I \text{ and } \sum \mu_i = 0; \\ & F_{\boldsymbol{c}} \in \boldsymbol{\mathcal{F}} \text{ and } F'_{\boldsymbol{c}}(0) = 1 \end{cases}$*

3 Sensitivity Analysis

In this section we study how sensitive the least squares and the semi-parametric estimate are to the initial guess $\hat{\boldsymbol{P}}$.

3.1 Sensitivity of Semi-Parametric Ranking

We now provide the sensitivity bounds for the semi parametric case. We will analyze problem (3) with respect to the variation of the input parameter $\hat{\boldsymbol{P}}$ to its true value $\underline{\boldsymbol{P}}$.

We will start by giving bounds on the true loss as a function of the empirical loss and the estimates provided by the solution to problem (3) and a sketch of the proof. This result is what we call the agnostic sensitivity analysis for it makes no assumption as to the true function F belonging to the set $\boldsymbol{\mathcal{F}}$, that is, it may be misspecified. It will also serve as a tool to prove the realizable case and therefore it constitutes one of our central results in the sensitivity analysis of the SPR.

Lemma 2 (Sensitivity Analysis Agnostic Case)**.** *Let the estimators be defined as above, then the difference between the true loss and the empirical loss is no more than a constant times $||\hat{\boldsymbol{P}} - \underline{\boldsymbol{P}}||$; that is:*

$$\left| \ ||\left(F_{\hat{\boldsymbol{c}}}^{-1}(\underline{\boldsymbol{P}}) - \Delta\hat{\boldsymbol{\mu}}\right)||_2 - \left|\left|\left(F_{\hat{\boldsymbol{c}}}^{-1}(\hat{\boldsymbol{P}}) - \Delta\hat{\boldsymbol{\mu}}\right)\right|\right|_2 \ \right| \leq \mathcal{L}_{\hat{c}} ||\hat{\boldsymbol{P}} - \underline{\boldsymbol{P}}||_2$$

where $\mathcal{L}_{\boldsymbol{c}} = |c_1| + 2|c_2| + ... + J|c_J| \leq J||\boldsymbol{c}||_1 \leq J \cdot U$

Proof (Sketch): $\left| \ ||\left(F_{\hat{\boldsymbol{c}}}^{-1}(\underline{\boldsymbol{P}}) - \Delta\hat{\boldsymbol{\mu}}\right)||_2 - \left|\left|\left(F_{\hat{\boldsymbol{c}}}^{-1}(\hat{\boldsymbol{P}}) - \Delta\hat{\boldsymbol{\mu}}\right)\right|\right|_2 \ \right|$

$$\leq_{[1]} ||F_{\hat{\boldsymbol{c}}}^{-1}(\underline{\boldsymbol{P}}) - F_{\hat{\boldsymbol{c}}}^{-1}(\hat{\boldsymbol{P}})||_2 \leq_{[2]} \sqrt{\sum_{ij} \mathcal{L}^2_{\hat{\boldsymbol{c}}} \left(\underline{P}_{ij} - \hat{P}_{ij}\right)^2}$$

[1] Every norm is $1-$Lipschitz continuous with respect to itself.

[2] $F_{\hat{\boldsymbol{c}}}^{-1}(p)$ is $\mathcal{L}_{\hat{\boldsymbol{c}}}-$Lipschitz continuous (to see this use the mean value theorem). □

Notice how this lemma shows the classical trade-off between the size of the hypothesis class (our parametric family of functions $\boldsymbol{\mathcal{F}}$), measured by $J \cdot U$, and the empirical loss over the respective hypothesis class. Also notice that no assumption was made here that the true F belongs to the set $\boldsymbol{\mathcal{F}}$, but in our case, the hypothesis class can arbitrarily approximate any distribution function at the cost of increasing $J \cdot U$. This lemma will also be used to prove the realizable case bounds that are analogous to the least squares sensitivity analysis to be provided in the following subsection.

Theorem 3 (Sensitivity Analysis Realizable Case)**.** *Let $\underline{\boldsymbol{P}}$ be such that $\underline{\boldsymbol{P}} = F_{\underline{\boldsymbol{c}}}(\Delta\underline{\boldsymbol{\mu}})$ for some $\underline{\boldsymbol{c}} \in \mathcal{C} \subset \mathbb{R}^J$ and $\underline{\boldsymbol{\mu}} \in \mathcal{M} \subset \mathbb{R}^I$. Also, define $\hat{\boldsymbol{P}}^* = F_{\hat{\boldsymbol{c}}}(\Delta\hat{\boldsymbol{\mu}})$, where $\hat{\boldsymbol{c}}$ and $\hat{\boldsymbol{\mu}}$ are the estimated parameters by formulation (3) and $||\cdot||$ be the L_2 norm; then:*

[4] Where the rows and columns of $\boldsymbol{Q^0}$ and $\boldsymbol{Q^1}$ are indexed from 0 to d and 0 to d-1 respectively.

1. *Empirical Loss Sensitivity:* $\left|\left|F^{-1}_{\hat{\boldsymbol{c}}}(\hat{\boldsymbol{P}}) - \Delta\hat{\boldsymbol{\mu}}\right|\right| \leq \mathcal{L}_{\underline{\boldsymbol{c}}}||\hat{\boldsymbol{P}} - \underline{\boldsymbol{P}}||$
2. *True Loss Sensitivity:* $\left|\left|F^{-1}_{\hat{\boldsymbol{c}}}(\underline{\boldsymbol{P}}) - \Delta\hat{\boldsymbol{\mu}}\right|\right| \leq (\mathcal{L}_{\underline{\boldsymbol{c}}} + \mathcal{L}_{\hat{\boldsymbol{c}}})\,||\hat{\boldsymbol{P}} - \underline{\boldsymbol{P}}||$

If we additionally assume $|\frac{d}{dx}F(\cdot)|$ *is upper-bounded by* $\mathcal{L}'$ *then:*
3. *Probability Matrix Sensitivity:* $\left|\left|\underline{\boldsymbol{P}} - \hat{\boldsymbol{P}}^*\right|\right| \leq \mathcal{L}'\,(\mathcal{L}_{\underline{\boldsymbol{c}}} + \mathcal{L}_{\hat{\boldsymbol{c}}})\,||\hat{\boldsymbol{P}} - \underline{\boldsymbol{P}}||$

If additionally $I + J \leq \frac{1}{2}I(I-1) + 3$ *and the set of equations* $F^{-1}_{\boldsymbol{c}}(\underline{\boldsymbol{P}}) = \Delta\boldsymbol{\mu}$ *on the parameters* $\boldsymbol{c}$ *and* $\boldsymbol{\mu}$ *together with the three additional constraints* $\sum \mu_i = 0$, $F(0) = 1/2$ *and* $F'(0) = 1$ *provide at least* $I + J$ *linearly independent constraints then:*
4. *Parameter Consistency:* $||\underline{\boldsymbol{c}} - \hat{\boldsymbol{c}}|| + ||\underline{\boldsymbol{\mu}} - \hat{\boldsymbol{\mu}}|| \longrightarrow 0$ *as* $\hat{\boldsymbol{P}} \longrightarrow \underline{\boldsymbol{P}}$

If we additionally assume that $\hat{\boldsymbol{P}}$ *obeys strong stochastic transitivity, that is,* $\hat{p}_{ij} < \hat{p}_{ji} \iff \hat{p}_{ik} < \hat{p}_{jk} \quad \forall k = 1, ..., n$ *then we have:*
5. *Order preservation:* $\hat{\mu}_{SP,i} < \hat{\mu}_{SP,j} \iff \hat{p}_{ij} < \hat{p}_{ji}$

where $\mathcal{L}_{\boldsymbol{c}} = |c_1| + 2|c_2| + ... + J|c_J| \leq J||\boldsymbol{c}||_1 \leq J \cdot U$.

Notice how the sensitivity constant for the probability matrix is upper-bounded by $J \cdot U$. This upper-bound is a measure of the degrees of freedom of the set in which function F belongs to; in fact it is proportional to the volume of the parameter space of the polynomial family. We will further on see a similar result for the least squares case. We will also show the relevance of these non-stochastic bounds on the quality of the refined matrix $\hat{\boldsymbol{P}}^*$, specifically we illustrate this in the following section by building on the recent results on estimation under the stochastic transitivity conditions for round robin tournaments.

3.2 Sensitivity of the Least Squares with Known F

Functions F_{BTL}, F_{THU}, and F_{THR} are Lipschitz continuous and the inverses F^{-1}_{BTL}, F^{-1}_{THU}, and F^{-1}_{THR} are also if we restrain the respective domains to non extreme cases $p \in [\epsilon, 1-\epsilon]$ (or equivalently $||\boldsymbol{\mu}|| \leq \mathcal{U} < \infty$). We therefore provide detailed sensitivity analysis for the non-extreme cases:

Theorem 4 (Sensitivity Analysis of the Least Squares Ranking Given F)**.** *Let* $\underline{\boldsymbol{P}} = F(\Delta\underline{\boldsymbol{\mu}})$ *be the underlying probability LST matrix given by a known* $\mathcal{L}-$*Lipschitz continuous function* F *and a unknown merit vector* $\underline{\boldsymbol{\mu}}$*. Given an estimate* $\hat{\boldsymbol{P}}$ *of* $\underline{\boldsymbol{P}}$ *such that* $\underline{p}_{ij}, \hat{p}_{ij} \in [\epsilon, 1-\epsilon]$ *for some fixed* $\epsilon \geq 0$*. Let* F^{-1} *be* $\mathcal{L}'$*-Lipschitz over* $[\epsilon, 1-\epsilon]$*, let* $I = n$ *be the number of players and let* $\hat{\boldsymbol{\mu}}_{LS}$ *be as in (1) and* $\hat{\boldsymbol{P}}^*_{LS} = F(\Delta\hat{\boldsymbol{\mu}}_{LS})$*; then:*

1. *Merit Sensitivity:* $||\hat{\boldsymbol{\mu}}_{LS} - \underline{\boldsymbol{\mu}}||_2 \leq \frac{\mathcal{L}'}{\sqrt{2n}}||\hat{\boldsymbol{P}} - \underline{\boldsymbol{P}}||_2$
2. *Probability Matrix Sensitivity:* $||\hat{\boldsymbol{P}}^*_{LS} - \underline{\boldsymbol{P}}||_2 \leq \mathcal{L}\cdot\mathcal{L}'||\hat{\boldsymbol{P}} - \underline{\boldsymbol{P}}||_2$

If we additionally assume that $\hat{\boldsymbol{P}}$ *obeys strong stochastic transitivity, that is,* $\hat{p}_{ij} < \hat{p}_{ji} \iff \hat{p}_{ik} < \hat{p}_{jk} \quad \forall k = 1, ..., n$ *then we have:*
3. *Order preservation:* $\hat{\mu}_{LS,i} < \hat{\mu}_{LS,j} \iff \hat{p}_{ij} < \hat{p}_{ji}$

Our bounds are non-stochastic, and therefore the probability matrix sensitivity must be tight (up to constant factors) for if not, one could refine the refined estimator indefinitely approximating the estimator to the true value. These sensitivity bounds serve as benchmarks for comparison with the semi-parametric case, for example, notice how $||\hat{\boldsymbol{P}}^* - \underline{\boldsymbol{P}}||_2$ is at most a constant away from $||\hat{\boldsymbol{P}} - \underline{\boldsymbol{P}}||_2$ whether F is known or not; similarly, the property of order preservation is maintained in both settings. At last, we wish to highlight that $\mathcal{L}\cdot\mathcal{L}'$ (the sensitivity analysis constant for the probability matrix) is a measure of the degrees of freedom of the class of functions in which the given function F belongs to; we have seen that the a similar bound is obtained for the semi-parametric case.

4 Statistical Properties

It should be clear by Theorems 3 and 4 that the consistency of the least squares refinement procedure and the semi-parametric refinement procedure stem solely from the realizability assumption and the consistency of the initial estimator $\hat{\boldsymbol{P}}$. In this section we will discuss the statistical properties of the least squares and the semi-parametric procedures as estimators; for this we must define beforehand how the estimate $\hat{\boldsymbol{P}}$ is obtained from the (summary) observable $\boldsymbol{W}$. We will illustrate how to use the sensitivity analysis developed in the previous section to analyze the quality of the overall estimation procedure with two cases:

1. Round Robin Case - In the round robin model we will refine the estimator derived in [4] which we will call $\hat{\boldsymbol{P}}_{\text{RSS+ISO}}$. The estimator there developed $\hat{\boldsymbol{P}}_{\text{RSS+ISO}}$ achieves the optimal minmax expected squared error bounds, it suffers though with the draw-back of being an exterior approximation (possibly infeasible). In this scenario, the main motivation is to obtain a feasible LST estimator of the probability matrix, recover the underlying function F and the merit $\boldsymbol{\mu}$ while retaining the optimal minmax rate of estimator $\hat{\boldsymbol{P}}_{\text{RSS+ISO}}$.
2. Blind Case - In this model we assume no particular structure in the matrix of paired comparisons $\boldsymbol{W}$. We here, due to the lack of structure, will use Laplace's estimator $\tilde{p}_{ij} = \frac{W_{ij}+1}{W_{ij}+W_{ji}+2}$ due to its stability and its Bayesian interpretation. Our main concern here is to provide an overall estimator that retains the properties of consistency and efficiency.

4.1 Round Robin Case

Theorems 1.1 and 1.2 of [4] characterizes a two step estimation procedure consistent with the strong stochastic transitivity condition, namely $\hat{\boldsymbol{P}}_{\text{RSS+ISO}}$. The results in [4], together with the results of [13], prove that $\hat{\boldsymbol{P}}_{\text{RSS+ISO}}$ is optimal in a min-max risk sense with mild conditions on the linear stochastic transitivity hypothesis.

The main drawback of this estimator, for the LST hypothesis, is that it can be infeasible. Even if the estimator happens to be feasible it doesn't fully enjoy the benefits of an LST model, that is, to be able to summarize the probability matrix with a merit vector and a distribution function. The recovery of function F is also in itself desirable for it allows to predict outcomes of games in the event of new players with new merit differences. To address these issues we will refine the estimator $\hat{\boldsymbol{P}}_{\text{RSS+ISO}}$ with our semi-parametric procedure and will give a result analogous to Theorems 1.1 and 1.2 of [4] for the LST case realizable by $\boldsymbol{\mathcal{F}}$:

Theorem 5. *Let* $\underline{\boldsymbol{P}}$ *be the underlying probability matrix of a paired comparison experiment and assume furthermore that* $I = n$ *and that matrix* $\boldsymbol{W}$ *obeys the round robin condition, that is,* $W_{ij} + W_{ji} = 1$ *for all* $i \neq j$*. Also let* $J \in \mathbb{N}$ *be an odd number and* $U < \infty$ *as in the definitions of Theorem 3.*

Let $\hat{\boldsymbol{P}}^*$ *be the refined estimator of* $\underline{\boldsymbol{P}}$ *defined by the three step procedure:*

Algorithm: RSS+ISO+SPR

Input: $\boldsymbol{W} \in \{0,1\}^{n\times n}$, $J \in \mathbb{N}$ *an odd number and* $U < \infty$.

1. *RSS (Row-Sum-Sort): Define a permutation* $\hat{\pi}_{RSS}$ *by sorting players by the number of victories (breaking ties randomly).*
2. *ISO (Isotonic Regression): Project matrix* $\hat{\pi}_{RSS}(\boldsymbol{W})$ *onto the isotonic cone (monotonically increasing values on both indexes) and then unsort the resulting matrix. (Producing* $\hat{\boldsymbol{P}}_{RSS+ISO}$*, for more details refer to [4].)*

3. *SPR (Semi-parametric Refinement): Refine the resulting estimator with the L_2 norm formulation given in problem (3).*

Output: $\boldsymbol{c} \in \mathbb{R}^{J+1}$, $\boldsymbol{\mu} \in \mathbb{R}^n$ *and* $\hat{\boldsymbol{P}}^* \equiv F_{\boldsymbol{c}}(\Delta\boldsymbol{\mu})$

Then there exists a universal constants $0 < K_7$ *such that:*

$$\sup_{\underline{\boldsymbol{P}} \in \mathbb{P}_{LST\text{-}\mathcal{F}}} \frac{1}{n^2}\mathbb{E}||\hat{\boldsymbol{P}}^* - \underline{\boldsymbol{P}}||_2^2 \leq K_7 \mathcal{L}' \frac{J^2 U^2 \log^2 n}{n} \tag{6}$$

Where K_7 *does not depend on neither* n *nor* J *and the set* $\mathbb{P}_{LST\text{-}\mathcal{F}}$ *is defined as the set of probability matrices realizable by functions* F *in the set* $\mathcal{F}$*. This retains the best worst case (min-max) expected squared error bound possible up to log factors.*

Proof (Sketch): The proof makes use of the probability matrix sensitivity analysis in Theorem 3 (topic 3) and Theorems 1 and 4 of [13] with Theorems 1.1 and 1.2 (read application (c) therein) of [4]. □

4.2 Blind Case

In the blind case we wish to provide an estimator that is consistent and efficient. Consistency of the refinement step, for the estimated probability matrix $\hat{\boldsymbol{P}}^*$, is a trivial consequence of the consistency of Laplace's rule of succession and the sensitivity bounds given by Theorem 3. In order to guarantee asymptotic efficiency of our estimator we add one more step to the process, we define the estimator $\bar{\boldsymbol{P}}^*$ as a three step procedure:

Algorithm: LAP+SPR+NEW

Input: $\boldsymbol{W} \in \{0,1\}^{n\times n}$, $J \in \mathbb{N}$ *an odd number and* $U < \infty$.

1. *LAP (Laplace's Estimator): Define an initial estimate* $\tilde{\boldsymbol{P}}$ *such that* $\tilde{P}_{ij} = \frac{W_{ij}+1}{W_{ij}+W_{ji}+2}$.
2. *SPR (Semi-parametric Refinement): Refine the resulting estimator with the formulation given in problem (3).*
3. *NEW (Newton Step): Perform one newton step on the likelihood equations to update the refined estimator.*

Output: $\boldsymbol{c} \in \mathbb{R}^{J+1}$, $\boldsymbol{\mu} \in \mathbb{R}^n$ *and* $\hat{\boldsymbol{P}}^* \equiv F_{\boldsymbol{c}}(\Delta\boldsymbol{\mu})$

All three steps have strait forward implementations (given that one has a SDP solver available for step 2). What is relevant for this case, is that we can now derive asymptotic consistency of our estimator $\bar{\boldsymbol{P}}^*$ in the blind case:

Corollary 2. *The estimator* $\bar{\boldsymbol{P}}^*$ *has asymptotic variance equal to the inverse Fisher information, and therefore it is the best possible achievable for consistent estimators.*

5 Numerical Experiments

In this section three numerical experiments illustrate our methodology . In our first numerical experiment we fit a seventh degree polynomial to the inverse of the BTL distribution function. The goal of this experiment is to verify if our semi-parametric model is capable of sufficiently approximating the BTL function with low degree polynomials. We performed a least squares fit of the function $F^{-1}_{\text{BTL}}(p) \equiv \log\left(\frac{p}{1-p}\right)$ within the range of probabilities from $p_{\min} = 0.01$ to $p_{\max} = .99$ with a grid of step size 0.01 (a total of 99 points). The result is displayed in Figure 1.

In our second experiment we generate 20 players with BTL merits sampled from a uniform distribution over the interval $[0, 10]$. These

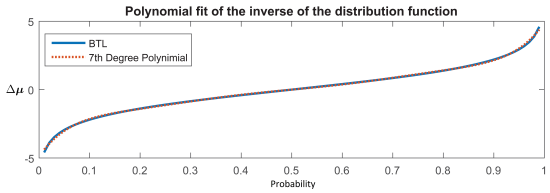


Figure 1: Polynomial fit of the inverse distribution function $F^{-1}_{\text{BTL}}(p)$.

players face each other in 50 random encounters generated with uniform probability over all pair of players. After every 5 games we estimate the underlying probability matrix $\boldsymbol{P} \in \mathbb{R}^{20\times 20}$ with Laplace's estimator $\tilde{\boldsymbol{P}}$ and compare it with the refined estimators. The refinement procedures used were the weighted and unweighted semi-parametric refinement procedure with $J = 5$ and the L_2 norm, the weighted and the unweighted least squares refinement procedure with the BTL function; in both weighed versions Ω_{ij} was chosen to be $W_{ij} + W_{ji}$. Figure 2 contains the average distance $||\hat{\boldsymbol{P}}^* - \boldsymbol{P}||_2$ of the resulting estimators and the underlying probability matrix after 20 runs of the described procedure.

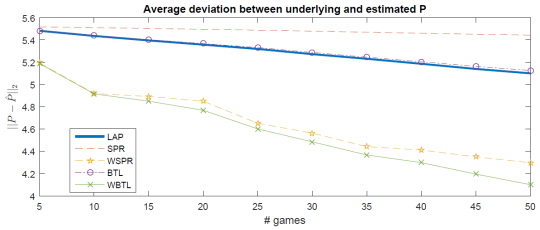


Figure 2: Comparison of refined estimators in the blind case.

In our second experiment we illustrate Theorem 5. In this setting we generate a sequence of round-robin paired comparison tournaments with an increasing number of players under a BTL model. We recover the underlying probability matrix using the RSS+ISO estimator and store the value of $\frac{1}{n^2}||\hat{\boldsymbol{P}} - \boldsymbol{P}||^2$ for the original estimator and for the refined estimators using polynomials of degree 3, 5 and 7. Figure 3 displays the average of $\frac{1}{n^2}||\hat{\boldsymbol{P}} - \boldsymbol{P}||^2$ after 30 runs for $n = 10, 20, 30, 40$ and 50.

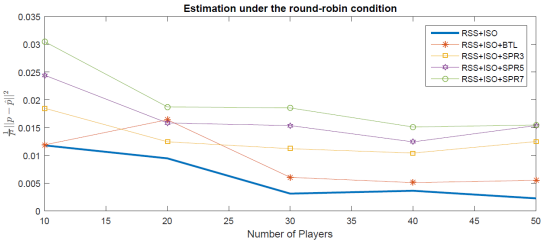


Figure 3: Refined estimators for round-robin tournaments.

Discussion of the numerical results As can be seen in Figure 1 the family of distribution functions with polynomial inverses approximates well the standard BTL model for J=7. The limit cases, when $p \to 0$ or $p \to 1$, though, are hard to approximate; this is so because $\lim_{p\to 0+} F^{-1}_{\text{BTL}}(p) = -\infty$ and $\lim_{p\to 1-} F^{-1}_{\text{BTL}}(p) = \infty$ and therefore the slight disturbances seen on the edges of the graphs depicted in Figure 1. The Thurstonian and the Threshold model also suffer from this hard limit case and in practice the polynomial approximation is not good close to the extremes. Though not depicted in Figure 1, polynomials of degree 3 and 5 also provide a good fit.

Our second experiment depicts the sensitivity analysis developed in Section 3. Our experiment tests Laplace's rule of succession as an estimator as well as the four refinement procedures described under the harsh condition of sparsity of matrix W. This condition is harsh because the number of paired comparisons is less than $\lceil 20 \log_2 20 \rceil = 87$ and the pairs compared were not chosen adaptively. Notice that under these conditions in Figure 2 the unrefined estimator is in fact slightly better than the unweighted BTL least squares and the semi-parametric refinement, but only by a constant factor as predicted by our sensitivity analysis. The weighted refinement procedures (with or without knowing F), though, perform much better than the unrefined estimator.

Our third experiment depicts Theorem 5 by showing the decrease in the average squared error with the increase of the number of players. Notice that the lower degree polynomials performed better than the higher degree polynomials; this is in accord with both the sensitivity analysis of Lemma 2 if we view it as an agnostic case and Theorem 3 if we view it as a realizable case, since the bounds increase with the increase of J.

6 Discussions

In this work we extended the least squares ranking estimation procedure to perform inference without the knowledge of the underlying distribution function F. We demonstrated that our inference procedure gives yield to a tractable SDP problem that can easily be incorporated as a refinement step to a given estimator of matrix $\boldsymbol{P}$. This refinement step may be used to infer simultaneously the underlying distribution function F and the player's merits $\boldsymbol{\mu}$ as well as a refined estimator $\hat{\boldsymbol{P}}^*$.

We provide non-stochastic sensitivity analysis of the standard least squares method as well as our semi-parametric procedure which guarantee that the distance $||\hat{\boldsymbol{P}}^* - \boldsymbol{P}||$ of our refined estimator to the true underlying probability matrix is at most a constant away from the distance $||\hat{\boldsymbol{P}} - \boldsymbol{P}||$ of the unrefined estimator to the underlying probability matrix. From our sensitivity analysis we also derive statistical properties of our resulting estimators such as consistency in the general case, min-max risk bounds in the round-robin case and efficiency with the addition of one Newton step.

In our numerical experiments we verified that the BTL distribution function is well approximated by the family of distribution functions with polynomial inverses. We also verified the sensitivity analysis bounds in two example cases, a round-robin scenario and the random sparse matches scenario.

Degenerate solutions Problem (5) can have multiple minima and degenerate solutions if not adjoined with additional constraints. Notice for example that the trivial solution (all null values) is not only feasible but also optimal. Another ill formulated problem that can occur is underdeterminacy (multiple solutions), to see this, consider the case where the statistician is provided with the exact value of $\underline{\boldsymbol{P}}$ and is informed that the probability matrix is realizable by $\boldsymbol{\mathcal{F}}$ for a given J, notice that if $J > I^2$ then the set of I^2 equations $F_c(\underline{p}_{ij}) = \mu_i - \mu_j$ would not suffice to determine the the $J + I$ parameters associated to this problem. To avoid a trivial solution and the problem of underdeterminacy one may consider adding equality constraints to the feasible set by fixing the value of function F or it's derivative F' at anchor points; other possibilities include constraining the merit space $\mathcal{M}$ with linear constraints or changing all together the initial estimate $\hat{\boldsymbol{P}}$. Notice though that if the statistician has reasons to believe that $J > I^2$ then a increase in the number of players I is imperative to make the parameters identifiable; the increase in the amount of games between players can increase the quality of the matrix $\hat{\boldsymbol{P}}$, but not of the parameters $\boldsymbol{c}$ and $\boldsymbol{\mu}$.

Future Research We can mention other questions that remain to be investigated in the field of linear stochastic transitivity relations. To begin with, little is known about the consequences of misspecification of function F in linear ranking models. Also we may inquire how to generalize the results in linear ranking models with unknown function F to incorporate multiplayer paired comparisons games with possibility of non-binary outcomes. At last our experiments indicate that the choice of the weights highly influence on the quality of the estimators derived in this article; we believe that this deserves further investigation and that this might reveal optimal procedures in the investigation of linear stochastic transitivity rankings.

7 Acknowledgments

The research leading to these results has received funding from the European Research Council under European Union's Horizon 2020 Program, ERC Grant agreement no. 682203 "SpeedInfTradeoff" and the research of Ori Davidov was partially supported by the Israeli Science Foundation Grant No. 1256/13.

REFERENCES

[1] Batchelder, William H.; Bershad, Neil J.; Simpson, Robert S. Dynamic paired-comparison scaling. *Journal of Mathematical Psychology*, (36): 185–212, 1992.

[2] Block, H.D.; Marschak, Jacob. Random orderings and stochastic theories of responses. *Cowles Foundation Discussion Paper*, (66), March 1959.

[3] Bradley, R. A.; Terry; M. E. . Rank analysis of incomplete block designs: I. the method of paired comparisons. *Biometrika*, (39), December 1952.

[4] Chatterjee, Sabyasachi ; Mukherjee, Sumit. On estimation in tournaments and graphs under monotonicity constraints. *arXiv:1603.04556 [math.ST]*, 2016.

[5] Csato, Laszlo. Additive and multiplicative properties of scoring methods for preference aggregation. *Corvinus Economics Working Papers*, 2014.

[6] deCani,John S. Maximum likelihood paired comparison ranking by linear programming. *Biometrika*, 3(56):537, 1969.

[7] Tom Minka; Graepel Thore; Ralf Herbrich;. Trueskill tm: A bayesian skill rating system. *Advances in Neural Information Processing Systems, MIT Press*, (20), January 2007.

[8] Huber, P.J. Pairwise comparison and ranking: optimum properties of the row sum procedure. *Annals of Mathematical Statistics*, 34(2):511–520, 1963.

[9] Xiaoye Jian; Lek-Heng Lim; Yuan Yao; Yinyu Ye. Statistical ranking and combinatorial hodge theory. *Mathematical Programming*, November 2010.

[10] Pablo A. Parrilo. Algebraic techniques and semidefinite optimization, 2016. URL `http://stellar.mit.edu/S/course/6/sp10/6.256/ /courseMaterial/topics/topic2/lectureNotes/ /lecture-10/lecture-10.pdf`.

[11] Regenwetter, Michel; Jason, Dana; Davis-Stober, C.P.; . Transitivity of preferences. *Psychological Review*, 118(1):42–56, 2011.

[12] Shah, Nihar; Balakrishnan, Sivaraman; Bradley, Joseph; Parekh, Abhay; Ramchandran, Kannan; Wainwright, Martin . Estimation from pairwise comparisons: sharp minimax bounds with topology dependence. *Proceedings of the Eighteenth International Conference on Artificial Intelligence and Statistics*, pages 856–865, 2015a.

[13] Shah, Nihar B.; Balakrishnan, Sivaraman; Guntuboyina, Adityanand. Stochastically transitive models for pairwise comparisons: statistical and computational issues. *arXiv:1510.05610 [stat.ML]*, 2015b.

[14] Thurstone, Louis L. A law of comparative judgment. *Psychology Review*, (34):273–286, 1927.

[15] Kristi Tsukida, ;Maya R. Gupta; How to analyze paired comparison data, May 2011. URL `www.dtic.mil/dtic/tr/fulltext/u2/a543806.pdf`.

[16] Yellott, J.I. The relationship between thurstone's, luce's and dawkins' models for paired comparisons. *Annual Meeting of Mathematical Psychology*, 1970.

Predicting diverse rankings in extreme multi-label classification

Kalina Jasinska[1] and **Krzysztof Dembczyński**[2]

Abstract. Extreme classification refers to multi-class and multi-label learning problems with very large output spaces, involving even millions of labels. This kind of problems brings many new challenges. One of them is the choice of the performance measure (loss function). Popular choices such as Hamming loss, precision@k or normalized discounted cumulative gain tend to favor the most popular labels (the most frequent) and ignore the long tail of the sparse ones. The reason of this behavior is that these measures rely directly on estimated posterior probabilities. In this paper, we discuss a different approach in which we will use relative scores of posterior probabilities, i.e., we rank the labels according to the difference between posterior probability and a label-specific threshold.

1 Introduction

Extreme classification refers to multi-class and multi-label learning problems involving hundreds of thousands [4] or even millions of labels [1]. In multi-class problems one and only one label is assigned to a training example. In the following, we focus on multi-label classification in which none, one, or more labels can be assigned to the example. It is often assumed, however, that the number of positive labels (i.e., those assigned to a training example) is small in comparison to the number of all possible labels.

Since in extreme classification we deal with such a large number of labels, we cannot expect that all labels will be properly checked and assigned to training examples. Therefore we often deal with unreliable training information and have to assume that among negative labels (those not assigned to an example) there may also be some relevant ones. Many labels will also be used very rarely by annotators. This, in turn, leads to the so-called long tail problem, i.e., a majority of labels is very sparse. In Figure 1, we show a distribution of label frequencies in one of recent English Wikipedia dumps (taken from https://dumps.wikimedia.org/backup-index.html). In this dataset, we have 1 827 532 categories (i.e., labels) and 12 301 697 articles. As we can see this distribution is characterized by a very long tail. In addition, the labels are also heavily imbalanced. The most popular category *living people* is assigned to less than 7% of articles.

Another specific issue of extreme classification is that the meaning of different labels can be very similar. For example, the article about Alan Turing has categories *English logicians* and *British logicians* assigned. Depending on the application one might want to predict both of these labels, or just one of them. In other words, we could prefer more diverse predictions, particularly, when only a limited number of predictions can be presented to the user.

[1] Institute of Computing Science, Poznań University of Technology, email: kjasinska@cs.put.poznan.pl
[2] Institute of Computing Science, Poznań University of Technology, email: kdembczynski@cs.put.poznan.pl

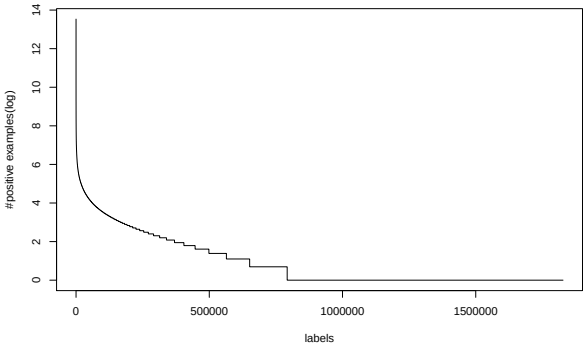


Figure 1. Frequencies on log scale of labels in the WikiLSHTC dataset

Taking the above into account, one of the most important questions in extreme classification is the choice of the right performance measure (or loss function) [2]. This measure should tackle all the issues coming with the problems at this scale. Unfortunately, many popular measures do not meet requirements of extreme classification. For example, such measures as Hamming loss, precision@k or normalized discounted cumulative gain (NDCG) favor the most popular labels (the most frequent) ignoring the long tail of sparse labels. In the following, we discuss an alternative approach that is based on relative scores which relate the estimated posterior probabilities to label-relative thresholds.

2 Limitations of traditional performance measures

Let $\boldsymbol{y} = (y_1, \ldots, y_m) \in \{0,1\}^m$ denote a vector of m ground truth labels (where $y_j = 1$ if and only if label j is relevant) assigned to an example $\boldsymbol{x} = (x_1, x_2, \ldots, x_p)$. In the extreme setting m is large, but the number of relevant labels is usually small. Below we consider Hamming loss and precision@k as examples of traditional performance measures in the extreme setting. As we will show they directly rely on estimated posterior probabilities which may favor the most popular labels. This is because of the fact that sparse labels are characterized not only by small prior probabilities $\eta_j = P(y_j = 1)$, but also by small posterior probabilities $\eta_j(\boldsymbol{x}) = P(y_j = 1|\boldsymbol{x})$.

The Hamming loss of a multi-label classifier $\boldsymbol{h}(\boldsymbol{x}) = (h_1(\boldsymbol{x}), h_2(\boldsymbol{x}), \ldots, h_m(\boldsymbol{x}))$, $h_j(\boldsymbol{x}) \in \{0,1\}$ is defined as:

$$\ell_H(\boldsymbol{y}, \boldsymbol{h}(\boldsymbol{x})) = \frac{1}{m} \sum_{j=1}^{m} [\![y_j \neq h_j(\boldsymbol{y})]\!] \,.$$

It equally weights false positives and false negatives. Therefore, the optimal strategy for this loss is $h_j^*(\boldsymbol{x}) = [\![\eta_j(\boldsymbol{x}) > 0.5]\!]$. In the extreme setting, usually only for a few labels the posterior estimates exceed the threshold of 0.5. Thus, the predictions will be very sparse.

ranking based on $\hat{\eta}_j(\boldsymbol{x})$					
score	label	$\hat{\eta}_j(\boldsymbol{x})$	$\hat{\eta}_j$	τ_j^F	y_j
0.5335	living people	0.534	0.4387	0.6081	0
0.1858	american male television actors	0.186	0.0073	0.2357	0
0.166	20th century american male actors	0.166	0.0057	0.1589	0
0.1475	american male film actors	0.147	0.0087	0.2276	0
0.1325	american film producers	0.133	0.0025	0.0736	1
ranking based on $\hat{\eta}_j(\boldsymbol{x}) - \eta_j$					
score	label	$\hat{\eta}_j(\boldsymbol{x})$	$\hat{\eta}_j$	τ_j^F	y_j
0.1785	american male television actors	0.188	0.0073	0.2357	0
0.1602	20th century american male actors	0.166	0.0058	0.1589	0
0.1389	american male film actors	0.147	0.0087	0.2276	0
0.1299	american film producers	0.133	0.0025	0.0736	1
0.1266	american film directors	0.130	0.0034	0.1162	1
ranking based on $\hat{\eta}_j(\boldsymbol{x}) - \tau_j^F$					
score	label	$\hat{\eta}_j(\boldsymbol{x})$	$\hat{\eta}_j$	τ_j^F	y_j
0.0589	american film producers	0.136	0.0025	0.0736	1
0.0221	american television producers	0.125	0.0020	0.1024	1
0.0138	american film directors	0.130	0.0034	0.1162	1
0.0114	20th century american business people	0.032	0.0016	0.0201	1
0.0082	american television personalities	0.041	0.0014	0.0329	0

Table 1. The top 5 predictions for the *Irwin Allen* article based on three different scores: $\hat{\eta}_j(\boldsymbol{x})$, $\hat{\eta}_j(\boldsymbol{x}) - \hat{\eta}_j$, and $\hat{\eta}_j(\boldsymbol{x}) - \tau_j^F$. The table reports scores, label names, $\hat{\eta}_j(\boldsymbol{x})$, $\hat{\eta}_j$, the macro F-measure optimized thresholds τ_j^F and the original labels.

For example, FastXML [5], a state-of-the-art classifier for extreme multi-label problems, trained on a subset of the Wikipedia dump (1 000 most frequent labels, 1 000 000 articles), would predict only one label for the article about *Irvin Allen*, if used with the threshold equal 0.5.

Instead of making binary prediction based on a threshold, we can ask for the top k labels according to the estimated posteriors $\hat{\eta}_j(\boldsymbol{x})$. In such a case, we are usually interested in optimizing precision@k, defined as

$$\text{precision@}k = \frac{1}{k}\sum_{r=1}^{k}[\![y_{\sigma(r)} = 1]\!]\,,$$

where $\sigma(r)$ gives an index of a label ranked on the r-th position.

3 Ranking based on label-relative scores

To overcome disadvantages of traditional performance measures in dealing with the long-tail problem, we follow a new approach that relies on label-relative scores. Instead of using the estimated posterior probabilities directly in making predictions, we advocate to use the difference between an estimate of the posterior probability, $\hat{\eta}_j(\boldsymbol{x})$, and a label specific threshold τ_j. In this case, a label i with a high posterior estimate can be ranked below a label j with a lower $\hat{\eta}_j(\boldsymbol{x})$, if the threshold for label i is relatively high making the difference $\hat{\eta}_i(\boldsymbol{x}) - \tau_i$ lower than the corresponding difference for label j.

Thresholds τ_j can be obtained in different ways. The simplest of them relies on using prior probabilities η_j. This corresponds to optimization of the so-called AM loss for each label separately [3]. Such a performance measure can be treated as a generalization of the Hamming loss with appropriately weighted false negatives and false positives. It has been shown in [3] that for many performance measures for binary classification (for example, the F-measure or Jaccard coefficient) the optimal strategy relies also on thresholding the posterior estimates. So, we can obtain the thresholds as a result of the optimization of a binary performance measure for each label separately. This approach is often referred to as macro-averaging of performance measures.

4 Example

In this section, we illustrate on a simple example the difference in predictions between the traditional approach based on estimates of posterior probabilities $\hat{\eta}_j(\boldsymbol{x})$ and the new approach based on label-relative scores. To this end, we trained FastXML on the aforementioned subset of the Wikipedia dump and estimated $\hat{\eta}_j(\boldsymbol{x})$ for an article about *Irwin Allen*. Then we computed three different types of scores: $\hat{\eta}_j(\boldsymbol{x})$, $\hat{\eta}_j(\boldsymbol{x}) - \hat{\eta}_j$, and $\hat{\eta}_j(\boldsymbol{x}) - \tau_j^F$, where τ_j^F is the macro F-measure optimized threshold. Table 1 presents the top 5 rankings based on these scores. In the used subset of the Wikipedia dump the distribution of labels is different than in the entire dump. For example, the *living people* label is much more frequent – its prior probability is 0.4387. Because of the high prior, this label was not included into the predictions based on label-relative scores (which are 0.0948 and -0.0746, respectively). So, the new approach was able to produce more diverse rankings and to push down the most popular labels. Moreover, the ranking based on the macro F-measure optimized thresholds pushed some negative labels with a high prior probability (*american male television actors*, *20th century american male actors*, *american male film actors*) down the ranking and pull the positive labels (*american film producers*, *american television producers*, *20th century american business people*, *american television personalities*) up, even if their posterior probabilities were lower than for other labels. Interestingly, the last label in the F-macro threshold based ranking has not been originally assigned as a relevant one to this article, but in fact it describes the article very well.

5 Conclusion

We investigated the problem of choosing the right performance measures for extreme classification in which we often deal with a long-tail of sparse labels. We advocated an approach in which we rank the labels according to the difference between an estimate of the posterior probability and a label-relative threshold.

Acknowledgments

This work has been supported by the National Science Centre under grant no. 2013/09/D/ST6/03917. Large-scale computations have been performed in Poznań Supercomputing and Networking Center.

REFERENCES

[1] R. Agrawal, A. Gupta, Y. Prabhu, and M. Varma, 'Multi-label learning with millions of labels: Recommending advertiser bid phrases for web pages', in *WWW*, pp. 13–24. ACM, (2013).

[2] H. Jain, Y. Prabhu, and M. Varma, 'Extreme multi-label loss functions for recommendation, tagging, ranking & other missing label applications', in *KDD*, (2016).

[3] Nagarajan Natarajan, Oluwasanmi Koyejo, Pradeep Ravikumar, and Inderjit S. Dhillon, 'Consistent binary classification with generalized performance metrics', in *Neural Information Processing Systems (NIPS)*, (2014).

[4] I. Partalas, A. Kosmopoulos, N. Baskiotis, T. Artières, G. Paliouras, É. Gaussier, I. Androutsopoulos, M.-R. Amini, and P. Gallinari, 'LSHTC: A benchmark for large-scale text classification', *CoRR*, (2015).

[5] Y. Prabhu and M. Varma, 'FastXML: A fast, accurate and stable tree-classifier for extreme multi-label learning', in *KDD*, pp. 263–272. ACM, (2014).

Optimizing the Generalized Gini Index in Multi-objective Bandits

Róbert Busa-Fekete[1] and **Paul Weng**[2] and **Orkun Karabasoglu**[2] and **Balázs Szörényi**[3 4]

Abstract. We study the multi-armed bandit (MAB) problem where the agent receives a vectorial feedback that encodes many possibly competing objectives to be optimized. The goal of the agent is to find a policy, which can optimize these objectives simultaneously in a fair way. This multi-objective online optimization problem is formalized by using the Generalized Gini Index (GGI) aggregation function. We propose two learning algorithms to tackle this task: one is a simple UCB-style algorithm, the other is a gradient descent-based algorithm that exploits the convexity of the GGI aggregation function. We test our algorithms on synthetic data as well as on an electric battery control problem where the goal is to trade off the use of the different cells of a battery in order to balance their respective degradation rates.

1 Introduction

The multi-armed bandit (MAB) problem (or bandit problem) refers to an iterative decision making problem in which an agent repeatedly chooses among K options, metaphorically corresponding to pulling one of K arms of a bandit machine. In each round, the agent receives a random payoff, which is a reward or a cost that depends on the arm being selected. The agent's goal is to optimize an evaluation metric, e.g., the *error rate* (expected percentage of times a suboptimal arm is played) or the *cumulative regret* (difference between the sum of payoffs obtained and the (expected) payoffs that could have been obtained by selecting the best arm in each round). In the *stochastic* multi-armed bandit setup, the payoffs are assumed to obey fixed distributions that can vary with the arms but do not change with time. To achieve the desired goal, the agent has to tackle the classical exploration/exploitation dilemma: It has to properly balance the pulling of arms that were found to yield low costs in earlier rounds and the selection of arms that have not yet been tested often enough [1, 2].

The bandit setup has become the standard modeling framework for many practical applications, such as online advertisement [3], medical treatment design [4], to name a few. In these tasks, the feedback is formulated as a single real value. However many real-world online learning problems are rather multi-objective. For example, in our motivating example, namely an electric battery control problem, the learner tries to discover a "best" battery controller, which balances the degradation rates of the battery cells (i.e., components of a battery), among a set of controllers while facing a stochastic power demand.

In this paper, we formalize the multi-objective multi-armed bandit setting in which the feedback received by the agent is in the form of a D-dimensional real-valued cost vector. The goal of the learning agent is to be both efficient, i.e., minimize the cumulative cost for each objective, and fair, i.e., balance the different objectives. The "fairness" between the objectives is quantified in terms of Generalized Gini Index (GGI), which is a well-known inequality measure in economics [5]. We propose two algorithms that extend two standard online methods widely used in single-objective optimization problems. In our synthetic and battery-control experiments we test them and demonstrate their versatility.

2 Formal setup

The multi-armed or K-armed bandit problem is specified by real-valued random variables $X_1, \dots, X_K$ associated, respectively, with K arms (that we simply identify by the numbers $1, \dots, K$). In each time step t, the online learner selects one and obtains a random sample of the corresponding distributions. These samples, which are called costs, are assumed to be independent of all previous actions and costs.[5] The goal of the learner can be defined in different ways, such as minimizing the sum of costs over time [2, 1].

In the *multi-objective* multi-armed bandit (MO-MAB) problem, costs are not scalar real values, but real vectors. More specifically, a D-objective K-armed bandit problem ($D \geq 2$, $K \geq 2$) is specified by K real-valued multivariate random variables $\mathbf{X}_1, \dots, \mathbf{X}_K$ over $[0,1]^D$. Let $\boldsymbol{\mu}_k = \mathbb{E}[\mathbf{X}_k]$ denote the expected vectorial cost of arm k where $\boldsymbol{\mu}_k = (\mu_{k,1}, \dots, \mu_{k,D})$. Furthermore, $\boldsymbol{\mu}$ denotes the matrix whose rows are the $\boldsymbol{\mu}_k$'s.

In each time step the learner can select one of the arms and obtain a sample, which is a cost vector, from the corresponding distribution. Sampling an arm is assumed to be independent over time and moreover independence also holds across the arms, but not necessarily across the components of cost vectors.

At time step t, k_t denotes the index of the arm played by a learner and $\mathbf{X}_{k_t}^{(t)} = (X_{k_t,1}^{(t)}, \dots X_{k_t,D}^{(t)})$ the resulting payoff. After playing t time steps, the empirical estimate of the expected cost $\boldsymbol{\mu}_k$ of the kth arm is

$$\widehat{\boldsymbol{\mu}}_k^{(t)} = \frac{1}{T_k(t)} \sum_{\tau=1}^{t} \mathbf{X}_{k_\tau}^{(\tau)} \mathbf{1}(k_\tau = k) \tag{1}$$

where all operations are meant elementwise, $T_k(t)$ is the number of times the kth arm has been played (i.e., $T_k(t) = \sum_{\tau=1}^{t} \mathbf{1}(k_\tau = k)$) and $\mathbf{1}(\cdot)$ is the indicator function.

[1] Paderborn University, email: busarobi@upb.de
[2] SYSU-CMU JIE, School of Electronics and Information Technology, SYSU-CMU JRI, email: {paweng,karabasoglu}@cmu.edu
[3] Technion Institute of Technology, email: szorenyi@inf.u-szeged.hu
[4] Research Group on AI, Hungarian Acad. Sci. and Univ. of Szeged

[5] Our setup is motivated by a practical application where feedback is more natural to formulate in terms of cost. However the stochastic bandit problem is most often formulated by using the notion of reward, which can be easily turn into cost by using the transformation $x \mapsto 1 - x$ assuming that the rewards are from $[0, 1]$.

In this study, we shall use the elementwise sample mean as an estimator of the mean vector. Nevertheless, it is worth to mention that this estimator is not admissible in general. For example, the James-Stein estimator [6] always achieves lower least square error in expectation than the elementwise sample mean given in (1) for D-dimensional normal distributions where $D \geq 3$ and the covariance matrix is diagonal in the form of σI. But to the best of our best knowledge, there does not exist a James-Stein estimator for such a general class of multivariate distributions like the class of distributions with bounded support that we focus on in this study.

3 Multi-objective optimization

In order to complete the MO-MAB setting, we need to introduce the notion of optimality of the arms. First, we define binary relation $\preceq$ on $\mathbb{R}^D$ as follows, for any $\boldsymbol{v}, \boldsymbol{v}' \in \mathbb{R}^D$:

$$\boldsymbol{v} \preceq \boldsymbol{v}' \Leftrightarrow \forall d = 1, \ldots, D, v_d \leq v'_d \ . \tag{2}$$

Let $\mathcal{O} \subseteq \mathbb{R}^D$ be a set of D-dimension real vectors. The *Pareto front* of $\mathcal{O}$, denoted $\mathcal{O}^*$, is the set of vectors such that:

$$\boldsymbol{v}^* \in \mathcal{O}^* \Leftrightarrow \left(\forall \boldsymbol{v} \in \mathcal{O}, \boldsymbol{v} \preceq \boldsymbol{v}^* \Rightarrow \boldsymbol{v} = \boldsymbol{v}^*\right) \ . \tag{3}$$

In multiobjective optimization, one usually wants to compute the Pareto front, or search for a particular element of the Pareto front. In practice, it may be costly (and even infeasible depending on the size of the solution space) to determine all the solutions of the Pareto front. One may then prefer to directly aim for a particular solution in the Pareto front. This problem is formalized as a mono-objective optimization problem, using an *aggregation function*.

An aggregation (or scalarizing) function, which is a non-decreasing function $\phi : \mathbb{R}^D \to \mathbb{R}$, allows every vector to receive a scalar value to be optimized. The initial multiobjective problem is then rewritten as follows:

$$\min \phi(\boldsymbol{v}) \quad \text{s.t.} \quad \boldsymbol{v} \in \mathcal{O} \ . \tag{4}$$

A solution to this problem yields a particular solution on the Pareto front. Note that if ϕ is not strictly increasing in every component, some care is needed to ensure that the solution of (4) is on the Pareto front.

Different aggregation function can be used depending on the problem at hand, such as sum, weighted sum, min, max, (augmented) weighted Chebyshev norm [7], Ordered Weighted Averages (OWA) [8] or Ordered Weighted Regret (OWR) [9] and its weighted version [10]. In this study, we focus on the Generalized Gini Index (GGI) [5], which is a special case of OWA.

4 Generalized Gini Index

The Generalized Gini Index (GGI) [5] is defined as

$$G(\mathbf{x}; \mathbf{w}) = \sum_{d=1}^{D} w_d x_d^{\downarrow}$$

where $\mathbf{x}^{\downarrow} = (x_1^{\downarrow}, \cdots, x_D^{\downarrow})$ contains the components of vector $\mathbf{x}$ that are sorted in a decreasing order and weights w_i's are non-increasing, i.e., $w_1 \geq w_2 \geq \ldots \geq w_D$. It is well-known that GGI is convex in $\mathbf{x}$ as it can be written as the maximum of linear functions.

GGI was originally introduced for quantifying the inequality of the welfare based on incomes. As an instance of Weighted Average Ordered Sample statistics, it has also been investigated in statistics [11]. The Weighted Average Ordered Sample statistics, also known as OWA [8], do not require that weights are non-increasing and are therefore not necessarily convex.

GGI has been characterized by Weymark [5]. It encodes both efficiency as it is monotone with Pareto dominance and fairness as it is non-increasing with Pigou-Dalton transfers [12, 13]. Informally, in our setting, a Pigou-Dalton transfer amounts to increasing an objective while decreasing another objective by the same quantity such that the order between the two objectives is not reversed. The effect of such a transfer is to balance a cost vector. Formally, GGI satisfies the following property: $\forall \mathbf{x}$ such that $x_i < x_j$,

$$\forall \epsilon \in (0, x_j - x_i), G(\mathbf{x} + \epsilon \boldsymbol{e}_i - \epsilon \boldsymbol{e}_j, \mathbf{w}) \leq G(\mathbf{x}, \mathbf{w})$$

where $\boldsymbol{e}_i$ and $\boldsymbol{e}_j$ are two vectors of the canonical basis. As a consequence, among vectors of equal sum, the best cost vector (w.r.t GGI) is the one with equal values in all objectives.

From now on, to simplify the presentation, we focus on GGI with strictly decreasing weights in $[0, 1]^D$, i.e., $d < d'$ implies $w_d > w_{d'}$. This means that GGI is strictly decreasing with Pigou-Dalton transfers. We also introduce a few notations. For a given $n \in \mathbb{N}$, $[n]$ denote the set $\{1, 2, \ldots, n\}$. Let $\mathbf{w}'$ be the vector defined by $\forall d \in [D], w'_d = w_d - w_{d+1}$ with $w_{D+1} = 0$. Note that all the components of $\mathbf{w}'$ are positive as we assume that those of $\mathbf{w}$ are strictly decreasing. Ogryczak and Sliwinski [14] showed that the GGI value of a vector $\mathbf{x}$ can be obtained by solving a linear program. We shall recall their results and define the linear program-based formulation of GGI.

Proposition 1. *The* GGI *score* $G(\mathbf{x}; \mathbf{w})$ *of vector* $\mathbf{x}$ *is the optimal value of the following linear program*

$$\begin{array}{lllll} \textit{minimize} & \displaystyle\sum_{d=1}^{D} w'_d \left(d r_d + \sum_{j=1}^{D} b_{j,d} \right) & & & \\ \textit{subject to} & r_d + b_{j,d} & \geq x_j & \forall j, d \in [D] \\ & b_{j,d} & \geq 0 & \forall j, d \in [D] \end{array}$$

Proof : The proof uses the Lorenz transform $L : \mathbb{R}^D \to \mathbb{R}^D$ which is defined as

$$L(\mathbf{x}) = \left(x_1^{\downarrow}, x_1^{\downarrow} + x_2^{\downarrow}, \cdots, \sum_{j=1}^{D} x_j^{\downarrow} \right) \ ,$$

and its dth component is denoted by $L_d(\mathbf{x})$. Then the GGI can be written as

$$\begin{aligned} G(\mathbf{x}; \mathbf{w}) &= \sum_{d=1}^{D} w_d x_d^{\downarrow} \\ &= \sum_{d=1}^{D} w_d (L_d(\mathbf{x}) - L_{d-1}(\mathbf{x})) \\ &= \sum_{d=1}^{D} (w_d - w_{d+1}) L_d(\mathbf{x}) \\ &= \sum_{d=1}^{D} w'_d L_d(\mathbf{x}) \end{aligned} \tag{5}$$

where we assume that $L_0(\mathbf{x}) = 0$.

For a given $d \in [D]$, $L_d(\mathbf{x})$ can also be thought of as the optimal solution of the linear program $\mathcal{P}_d$:

$$\begin{array}{lll} \text{maximize} & \sum_{j=1}^{D} y_j x_j & (= L_d(\mathbf{x})) \\ \text{subject to} & \sum_{j=1}^{D} y_j = d & \\ & y_j \in [0,1] & \end{array} \tag{6}$$

To see this, first note that $y_1, \ldots, y_D$ represent decision variables. Thus in principle they should be in $\{0,1\}$, i.e., $y_j = 1$ for an optimal solution means that the jth component of $\mathbf{x}$ is among the d biggest values of $\mathbf{x}$. A simple argument by contradiction shows that optimal solutions are attained at extreme points of $[0,1]^D$.

The dual $\mathcal{D}_d$ of the above problem $\mathcal{P}_d$ is:

$$\begin{array}{lllll} \text{minimize} & dr_d + \sum_{j=1}^{D} b_{j,d} & (= L_d(\mathbf{x})) & & \\ \text{subject to} & r_d + b_{j,d} & \geq x_j & & \forall j \in [D] \\ & b_{j,d} & \geq 0 & & \forall j \in [D] \end{array} \tag{7}$$

According to the strong duality theorem, its optimal solution equals to the optimal solution of the primal problem, which is $L_d(\mathbf{x})$ by construction. This, together with Equation (5) yields the proposition's claim. ■

Interestingly, when realizable solution $\mathbf{x}$ belongs to a polytope, using the linear programs of (7) leads to formulate a simple linear program to solve, which consists in adding the polytope constraints to the linear program of Proposition 1. However, the matter is not as simple if one were to use the linear programs of (6). First, the optimization of GGI and that of (7) are in opposite direction. Second, the resulting optimization program would be quadratic.

5 Optimal policy

In the mono-objective case, the arms are compared in terms of their means, which induces a ranking over the arms, and thus the notion of the best arm is also well-defined. In the multi-objective setting, we make use of the notion of the GGI criterion to compare arms. One can compute the GGI score of each arm k as $G(\boldsymbol{\mu}_k; \mathbf{w})$ if its vectorial mean $\boldsymbol{\mu}_k$ is known. Then the optimal arm is the one that minimizes the GGI score, i.e.,

$$k^* = \operatorname*{argmin}_{k \in [K]} G(\boldsymbol{\mu}_k; \mathbf{w}) \ .$$

Since the GGI operator is convex, $G(\boldsymbol{\mu}_k; \mathbf{w}) = G(\mathbb{E}[\mathbf{X}_k]; \mathbf{w}) \leq \mathbb{E}[G(\mathbf{X}_k; \mathbf{w})]$ by Jensen's inequality.

We are going to deal with mixed strategies, which can be defined as $\mathcal{A} = \{\boldsymbol{\alpha} \in \mathbb{R}^K | \sum_{k=1}^{K} \alpha_k = 1 \wedge 0 \preceq \boldsymbol{\alpha}\}$, because they may allow to reach lower GGI values than any fixed arm. A policy, which is parameterized by $\boldsymbol{\alpha}$ chooses arm k with probability α_k. The optimal mixed policy can be computed as

$$\boldsymbol{\alpha}^* = \operatorname*{argmin}_{\boldsymbol{\alpha} \in A} G\left(\sum_{k=1}^{K} \alpha_k \boldsymbol{\mu}_k; \mathbf{w}\right) \ . \tag{8}$$

In general, $G\left(\sum_{k=1}^{K} \alpha_k^* \boldsymbol{\mu}_k; \mathbf{w}\right) \leq G(\boldsymbol{\mu}_{k^*}; \mathbf{w})$, therefore using mixed strategies is justified in our setting. Recalling Proposition 1, it is clear that solving the following linear program

$$\begin{array}{lll} \text{minimize} & \sum_{d=1}^{D} w'_d \left(dr_d + \sum_{j=1}^{D} b_{j,d} \right) & \\ \text{subject to} & r_d + b_{j,d} \geq \sum_{k=1}^{K} \alpha_k \mu_{k,j} & \forall j, d \in [D] \\ & \boldsymbol{\alpha}^T \mathbf{1} = 1 & \\ & \boldsymbol{\alpha} \geq \mathbf{0} & \\ & b_{j,d} \geq 0 & \forall j, d \in [D] \end{array} \tag{9}$$

amounts to solving (8).

6 Regret

After playing T time steps, the average cost of this learner can be written as

$$\mathbf{X}^{(T)} = \frac{1}{T} \sum_{t=1}^{T} \mathbf{X}_{k_t}^{(t)} \ .$$

Our goal is to minimize the GGI index of this term. Accordingly we expect the learner to collect costs so as their average in terms of GGI, that is, $G\left(\mathbf{X}^{(T)}; \mathbf{w}\right)$ should be as small as possible. As shown in the previous section, for a given bandit instance with arm means $\boldsymbol{\mu} = (\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_K)^\intercal$, the optimal policy $\boldsymbol{\alpha}^*$ achieves $G\left(\sum_{k=1}^{K} \alpha_k^* \boldsymbol{\mu}_k; \mathbf{w}\right) = G\left(\boldsymbol{\alpha}^{*\intercal} \boldsymbol{\mu}; \mathbf{w}\right)$ if the randomness of the costs are not taken into account. We consider the performance of the optimal policy as a reference value, and define the regret of the learner as the difference of the GGI of its average cost and the GGI of the optimal policy:

$$R^{(T)} = G\left(\mathbf{X}^{(T)}; \mathbf{w}\right) - G\left(\boldsymbol{\alpha}^{*\intercal} \boldsymbol{\mu}; \mathbf{w}\right) \ .$$

Note that the GGI is a continuous function, therefore if the learner follows a policy $\boldsymbol{\alpha}^{(T)}$ that is "approaching" to $\boldsymbol{\alpha}^*$ as $T \to \infty$, then the regret is vanishing. In this paper, we are interested in the expected regret $\mathbb{E}\left[R^{(T)}\right]$ where the expectation is meant with respect to the randomization of the learner and on the randomness of the costs.

7 Learning algorithms

In this section we propose two algorithms, which can optimize the regret defined in the previous section. The first one is devised based on the principle of "optimism in the face of uncertainty" [1], that is, the algorithm makes its decision based on the optimistic estimate of the arm means. The second algorithm exploits the convexity of the GGI operator and formalizes the policy search problem as an online convex optimization problem, which is solved by a gradient descent algorithm with projection [15].

7.1 Optimistic algorithm

The principle of "optimism in the face of uncertainty" is very general and applies to many bandit problems where the environment is stochastic. Assume a learner who has already observed costs on various arms. Based on the observations, the mean of the arm distributions can be estimated with some precision. Then the learner selects the next arm based on the optimistic estimates, that is, the best possible estimates, which is in our setting the high probability lower confidence bounds.

This principle can be easily adapted to our multi-objective setup as well. As mentioned before, with the knowledge of the arm means $\boldsymbol{\mu} = (\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_K)^\intercal$, the optimal policy can be computed by solving the linear program given in (9). The idea of our algorithm is to solve the same linear program but the arm means are replaced by their optimistic estimates. That is, the learner solves the following linear program in each time step t:

$$\begin{aligned} \text{minimize} \quad & \sum_{d=1}^{D} w'_d \left(dr_d + \sum_{j=1}^{D} b_{j,d} \right) \\ \text{subject to} \quad & r_d + b_{j,d} \geq \sum_{k=1}^{K} \alpha_k \left(\widehat{\mu}_{k,j}^{(t)} - c_k^{(t)} \right) \ \ \forall j, d \in [D] \\ & \boldsymbol{\alpha}^T \mathbf{1} = 1 \\ & \boldsymbol{\alpha} \geq \mathbf{0} \\ & b_{j,d} \geq 0 \qquad \forall j, d \in [D] \end{aligned}$$

where $\widehat{\boldsymbol{\mu}}_k^{(t)} = \left[\widehat{\mu}_{k,j}^{(t)} \right]_{1 \leq j \leq D}$ is the mean estimate of kth arm based on the observed costs up to time t and $c_k^{(t)}$ is the confidence interval defined as

$$c_k^{(t)} = \sqrt{\frac{2 \log t}{T_k(t)}} \ .$$

The confidence interval was motivated by the UCB algorithm [1]. We refer to this algorithm as OPTIMISTIC.

7.2 Gradient descent

The multi-objective regret optimization problem can be viewed as a convex optimization task, since the GGI function $G(\mathbf{x}; \mathbf{w})$ is convex in $\mathbf{x}$. Moreover, with the precise knowledge of the arm means $\boldsymbol{\mu} = (\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_K)^\intercal$, one may compute the gradient of $G(\boldsymbol{\alpha}^\intercal \boldsymbol{\mu}; \mathbf{w})$ with respect of $\boldsymbol{\alpha}$, which allows us to apply some online convex optimization technique [16] to tackle this online learning problem. In this section, we devise an algorithm that is motivated by this observation and that uses the empirical estimates of the arm means.

We shall make use of a well-known online convex optimization algorithm called ADAGRAD [17] that computes the gradient step in an adaptive way by scaling the length of the descent step based on the gradients observed in the previous rounds. In each round, the ADAGRAD algorithm first takes a gradient descent step and then project the new point back to the feasible domain. The domain of our optimization problem is $\mathcal{A} = \{\boldsymbol{\alpha} \in \mathbb{R}^K | \sum_{k=1}^K \alpha_k = 1 \wedge 0 \preceq \boldsymbol{\alpha}\}$, which is a convex set. First, assume a policy $\boldsymbol{\alpha}$ whose GGI is $G(\boldsymbol{\alpha}^\intercal \boldsymbol{\mu}; \mathbf{w})$. The gradient of GGI with respect to α_i can be computed as

$$\frac{\partial G(\boldsymbol{\alpha}^\intercal \boldsymbol{\mu}; \mathbf{w})}{\partial \alpha_k} = \sum_{d=1}^{D} w_d \mu_{k,\pi(d)} \ . \tag{10}$$

where π is the permutation that sorts the components of $\boldsymbol{\alpha}^\intercal \boldsymbol{\mu}$ in a decreasing order. Means $\boldsymbol{\mu}_k$'s are not known but they can be estimated based on the costs observed so far.

The multi-objective gradient descent algorithm is defined in Algorithm 1, which we shall refer to as MO-ADAGRAD. The algorithm first pulls each arm at once as an initialization step. Then in each iteration, it computes an estimate of $\boldsymbol{\alpha}^{(t)\intercal} \boldsymbol{\mu}$, which is the expected cumulative cost of the current policy $\boldsymbol{\alpha}^{(t)}$ (line 6). This estimate is used to sort the objectives in a decreasing order, which we need to compute the gradient estimate of the GGI with respect to the current policy $\boldsymbol{\alpha}^{(t)}$ given in (10). As a next step, the algorithms takes

Algorithm 1 MO-ADAGRAD (η)

1: **for** rounds $t = 1 \to K$ **do**
2: set $k_t = t$ and $T_{k_t} = 1$
3: pull arm k_t and observe sample $\mathbf{X}_{k_t}^{(t)}$
4: Set $\boldsymbol{\alpha}^{(K)} = (1/K, \cdots, 1/K)$ and $\boldsymbol{G}^{(K)} = \mathbf{0}$
5: **for** rounds $t = K+1, K+2, \ldots$ **do**
6: $\mathbf{v}^{(t)} = \boldsymbol{\alpha}^{(t)\intercal} \widehat{\boldsymbol{\mu}}^{(t)}$ ▷ $\mathbf{v}^{(t)} \approx \boldsymbol{\alpha}^{(t)\intercal} \boldsymbol{\mu}$
7: $\pi = \text{argsort}\left(\mathbf{v}^{(t)}\right)$ ▷ Permutation for gradient estimate
8: **for** $k = 1 \to K$ **do** ▷ Compute the gradient estimate
9: $g_k^{(t)} = 0$
10: **for** $d = 1 \to D$ **do**
11: $g_k^{(t)} = g_k^{(t)} + w_d \widehat{\boldsymbol{\mu}}_{k,\pi(d)}^{(t)}$
12: $\boldsymbol{G}^{(t)} = \boldsymbol{G}^{(t-1)} + \boldsymbol{g}^{(t)\intercal} \boldsymbol{g}^{(t)}$ ▷ Gradient step
13: $\hat{\boldsymbol{\alpha}}^{(t)} = \boldsymbol{\alpha}^{(t-1)} - \eta {\boldsymbol{G}^{(t)}}^{-1/2} \mathbf{g}^{(t)}$
14: $\boldsymbol{\alpha}^{(t)} = \text{argmin}_{\boldsymbol{\alpha} \in \mathcal{A}} \|\boldsymbol{\alpha} - \hat{\boldsymbol{\alpha}}^{(t)}\|_{G^{(t)1/2}}$ ▷ Projection
15: Choose an arm k_t according to $\boldsymbol{\alpha}^{(t)}$
16: Set $T_{k_t} = T_{k_t} + 1$
17: Pull arm k_t, and observe sample $\mathbf{X}_{k_t}^{(t)}$

the gradient step (line 12) according to the ADAGRAD algorithm and carries out the projection step where

$$\|\boldsymbol{\alpha} - \hat{\boldsymbol{\alpha}}^{(t)}\|_{\boldsymbol{G}} = \left(\boldsymbol{\alpha} - \hat{\boldsymbol{\alpha}}^{(t)}\right)^\intercal \boldsymbol{G} \left(\boldsymbol{\alpha} - \hat{\boldsymbol{\alpha}}^{(t)}\right) \ .$$

The rationale behind this gradient step and projection is that the value of the multi-variate function to be optimized might be changing faster in some directions resulting in larger step concerning this direction, and the step size, of course, depends on the value of the gradient. This kind of diversity among the step size of various directions is taken into account in the projection step by scaling the distance by the inverse of the cumulative gradient ${G^{(t)}}^{1/2}$. Finally, the MO-ADAGRAD chooses an arm according to $\boldsymbol{\alpha}^{(t)}$ and observes the vectorial cost.

The algorithm has only one hyperparameter, which is the step size η. We found this algorithm robust to this hyperparameter, which might be explained by its adaptive nature. We always set the value of η to 0.1 in our experiments.

8 Experiments

To test our algorithms, we carried out two sets of experiments. In the first we generated synthetic data from multi-objective bandit instances with known parameters. In this way, we could compute the regret defined in Section 6 and, thus investigate the empirical performance of the algorithms. In the second set of experiments, we run our algorithm on a complex multi-objective online optimization problem, namely an electric battery control problem.

8.1 Synthetic Experiments

We generated random multi-objective bandit instances for which each component of the multivariate cost distributions obeys Bernoulli distribution with various parameters. The parameters of each Bernoulli distributions are drawn uniformly at random from $[0, 1]$ independently from each other. The number of arms K was set to 10 and the dimension of the reward distribution was taken from $D \in \{2, 5, 10, 20\}$. The weight vector $\mathbf{w}$ of GGI was set to $w_d = 1/2^{d-1}$. Since the parameters of the bandit instance are known, we could

compute the regret defined in Section 6. We ran the MO-ADAGRAD and OPTIMISTIC algorithms with 20 repetitions. The multi-objective bandit instance were regenerated after each run. The regrets of the two algorithms, which are averaged out over the repetitions, are plotted in Figure 1. The results reveal some general trends. First, the average regrets of both algorithms converge to zero. Second the OPTIMISTIC algorithm outperforms the gradient descent algorithm for small number of round, typically $T < 5000$. This fact might be explained by the fact that the optimistic algorithm solves a linear program for estimating α^* whereas the MO-ADAGRAD minimizes the same objective but using a gradient descent approach with projection, which might achieve slower convergence in terms of regret, but its computational time is significantly decreased compared to the optimistic algorithm.

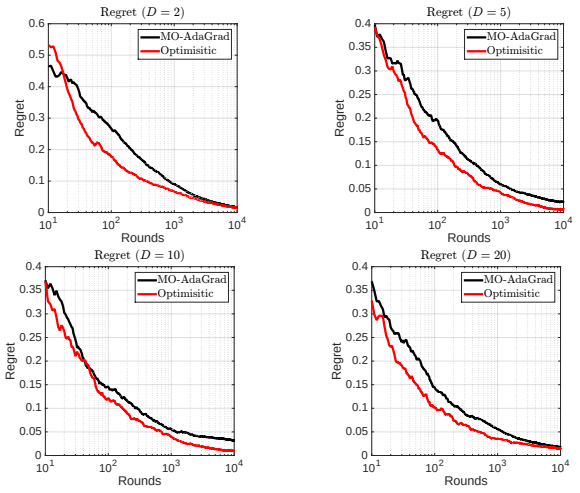


Figure 1. The regret of the OPTIMISTIC and MO-ADAGRAD. The regret is averaged over 20 repetitions and plotted in terms of the number of rounds. The dimension of the arm distributions was set to $D \in \{2, 5, 10, 20\}$, which is indicated in the title of the panels.

8.2 Battery control task

Efficient management of electric batteries leads to better performance and longer battery life, which is important for instance for electric vehicles whose range is still limited compared to those with petrol or diesel engines. An electric battery is composed of several cells whose capacity varies extensively due to inconsistencies in weight and volume of the active material in their individual components, different internal resistance and higher temperatures leading to different aging rates. As a consequence, at any instant, the energy output is different from each cell, which ultimately results in faster rates of decay and ultimately leads to the failure of the battery. To address this problem, a control strategy called cell balancing is utilized, which aims at maintaining a constant energy level — mainly state-of-charge (SOC) — in each cell, while controlling for temperature and aging. Many cell-balancing controllers can be defined, depending on the importance given to the three objectives: SOC, temperature and aging. The values of those objectives should be balanced between the cells because a balanced use of all cells leads to a long lasting system. Moreover, those objectives values should also be balanced within a cell, because for example, a cell can have higher capacity on a higher temperature, but at the same time it has a higher risk to explode.

Our battery control task consists in learning and selecting the "best" cell balancing in an online fashion, so that it can dynamically adapt to the consumption profile and environment, such as outdoor temperature. In case of electric cars, this means that the controller needs to be able to adapt to the driving habits of the driver and to the terrain, such as hilly roads or desert roads. In this experiment, our goal was more specifically to test our multi-objective online learning algorithms in the battery control task, and verify that our online learning algorithms can indeed find a policy for this control task which leads to a balanced parameter values of the cells.

The battery is modeled using the internal resistance (Rint) model [18]. The estimation of SOC is based on the Ampere Hour Counting method [19]. The variation of temperature in the system is determined according to the dissipative heat loss due to the internal resistance and thermal convection [20]. Cell degradation or aging is a function of temperature, charging/discharging rates and cumulative charge [21]. Moreover, the battery model is complemented with 10 different cell-balancing controllers. The whole model is implemented in the Matlab/Simulink software package and can emulate any virtual situation whose electric consumption is described by a time series, which is given as input to the model. For a chosen controller, the output of the model comprises of the objective values of each battery cell at the end of the simulation. Note that the output of the battery model is a multivariate random vector since the power demand is randomized, therefore this control task can be readily accommodated into our setup. In our experiments, the battery consists of 4 cells, thus $D = 12$ in this case. The cell-balancing controllers correspond to the arms, thus $K = 10$.

The online learning task consists of selecting among these controllers/arms so that the final objective values are as balanced as possible. The experiment was carried out as follows: in each iteration, the learner selects a controller according to its policy, then the battery model is run with the selected controller by using a random consumption time series. At the end of the simulation, the learner receives the objective values of each cell output by the battery model as feedback and updates its policy. The goal of the learner is to find a policy over the controllers, which leads to a balanced state of cells in terms of cumulative value.

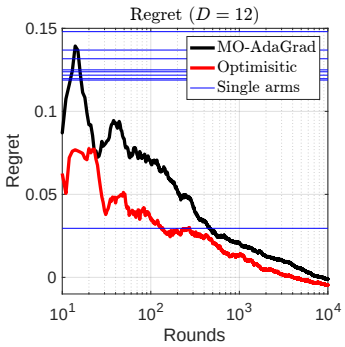


Figure 2. The regret of the OPTIMISTIC and MO-ADAGRAD on the battery control task. The regret is averaged over 10 repetitions and plotted in terms of the number of rounds. The dimension of the arm distributions was $D = 12$.

The results are shown in Figure 2. In this experiments we do not know the means of the arms, but we estimated them based on 30 runs. These mean estimates were used for computing the optimal policy and the regret. We run the MO-ADAGRAD and OPTIMISTIC

over 10 repetitions. Their average regret exhibits the same trend like in the synthetic experiments: the OPTIMISTIC achieved faster convergence. The blue lines shows the regret of the pure policies, which selects always the same arm, i.e., the performance of single strategies. It is important to mention that the optimal mixed controller has lower GGI value since the regret of any arm is positive, and more importantly, the two learners converge to optimal mixed policies in terms of regret. Note that the regret for larger number of time steps is slightly negative, which stems from the fact that we estimated the means of the arms (because their true values are not known).

9 Related work

Multi-armed bandit problems have generated significant theoretical interest, and they have been applied to many real applications [22, 23]. The single-objective MAB problem has been intensively studied especially in recent years, nevertheless there is only a very limited number of work concerning the multi-objective setting. To the best of our best knowledge, Drugan and Nowé [24] considered first the multi-objective multi-armed problem in a regret optimization framework with a stochastic assumption. Their work consists of extending the UCB algorithm [1] so as to be able to handle multi-dimensional feedback vectors with the goal of determining all arms on the Pareto front. Their algorithm makes use of the optimistic estimate of the arm means like our algorithm, but it chooses an arm uniformly at random from the set of arms whose optimistic estimate is on the Pareto front. Their regret definition is based on the distance between the mean of the arms and the Pareto front. Therefore contrary to our regret notion, their regret does not encode fairness.

Azar et al. [25] investigated a sequential decision making problem with vectorial feedback. In their setup the agent is allowed to choose from a finite set of actions and then it observes the vectorial feedback for each action, thus it is a full information setup whereas our setup is a bandit information setting because the agent observes only the feedback corresponding to the chosen arm. Moreover, the feedback is non-stochastic in their setup, as it is chosen by an adversary. They propose an algorithm which can handle a general class of aggregation function, such as the set of bounded domain, continuous, Lipschitz and quasi-concave functions.

10 Conclusion and future work

We introduced a new problem in the context of multi-objective multi-armed bandit (MOMAB). Contrary to most previously proposed approaches in MOMAB, we do not search for the Pareto front, instead we aim for a fair solution, which is important for instance when each objective corresponds to the payoff of a different agent. To encode fairness, we use the well-known generalized Gini index (GGI), a criterion developed in economics. To optimize this criterion, we proposed two algorithms, one based on the principle of "optimisim in the face of uncertainty" and the other exploiting the convexity of GGI. We evaluated both algorithms on two domains and obtained promising experimental results. First, we validated our propositions on random instances of MOMAB. Then, we tried the two algorithms on a more realistic task, which is an electric battery control problem.

As future work, a theoretical analysis is needed for the two algorithms we proposed. Indeed, we plan to prove regret bounds for them and obtain matching lower bounds. Moreover, while we focused in this paper on the stochastic setting, it would be worthwhile to extend our work to the adversarial setting. Finally, it would also be interesting to extend this work to the contextual multi-armed bandit and/or the reinforcement learning setting, which would be useful to solve the electric battery control problem even more finely.

REFERENCES

[1] P. Auer, N. Cesa-Bianchi, and P. Fischer. Finite-time analysis of the multiarmed bandit problem. *Machine Learning*, 47:235–256, 2002.

[2] T. L. Lai and Herbert Robbins. Asymptotically efficient adaptive allocation rules. *Advances in Applied Mathematics*, 6(1):4–22, 1985.

[3] A. Slivkins. Contextual bandits with similarity information. *J. Mach. Learn. Res.*, 15:2533–2568, 2014.

[4] W.H. Press. Bandit solutions provide unified ethical models for randomized clinical trials and comparative effectiveness research. In *Proceedings of the National Academy of Sciences*, volume 106, pages 22398–22392, 2009.

[5] John A. Weymark. Generalized gini inequality indices. *Mathematical Social Sciences*, 1(4):409 – 430, 1981.

[6] William James and Charles Stein. Estimation with quadratic loss. In *Proceedings of the fourth Berkeley symposium on mathematical statistics and probability*, volume 1, pages 361–379, 1961.

[7] R.E. Steuer and E.-U. Choo. An interactive weighted Tchebycheff procedure for multiple objective programming. *Mathematical Programming*, 26:326–344, 1983.

[8] R.R. Yager. On ordered weighted averaging aggregation operators in multi-criteria decision making. *IEEE Trans. on Syst., Man and Cyb.*, 18:183–190, 1988.

[9] W. Ogryczak, P. Perny, and P. Weng. On minimizing ordered weighted regrets in multiobjective Markov decision processes. In *International Conference on Algorithmic Decision Theory (ADT)*, Lecture Notes in Artificial Intelligence. Springer, 2011.

[10] W. Ogryczak, P. Perny, and P. Weng. A compromise programming approach to multiobjective Markov decision processes. *International Journal of Information Technology & Decision Making*, 12:1021–1053, 2013.

[11] Zoltán Buczolich and Gábor J. Székely. When is a weighted average of ordered sample elements a maximum likelihood estimator of the location parameter? *Advances in Applied Mathematics*, 10(4):439 – 456, 1989.

[12] A. Pigou. *Wealth and Welfare*. Macmillan, 1912.

[13] H. Dalton. The measurement of inequality of incomes. *Economic J.*, 30(348–361), 1920.

[14] W. Ogryczak and T. Sliwinski. On solving linear programs with the ordered weighted averaging objective. *Eur. J. Operational Research*, 148:80–91, 2003.

[15] Martin Zinkevich. Online convex programming and generalized infinitesimal gradient ascent. In *ICML 2003*, 2003.

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

[17] John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *J. Mach. Learn. Res.*, 12:2121–2159, July 2011.

[18] V.H. Johnson. Battery performance models in ADVISOR. *Journal of Power Sources*, 110:312–329, 2002.

[19] J. Dambrowski. Review on methods of state-of-charge estimation with viewpoint to the modern $LiFePO_4$/$Li_4Ti_5O_{12}$ lithium-ion systems. In *International Telecommunication Energy Conference*, 2013.

[20] Lijun Gao, Shenyi Chen, and Roger A. Dougal. Dynamic lithium-ion battery model for system simulation. *IEEE Trans. on Components and Packaging Technologies*, 25(3):495–505, 2002.

[21] Gao Tao. Research on $LiMn_2O_4$ battery's state of charge estimation with the consideration of degradation. Technical report, Tsinghua University, 2012.

[22] N. Cesa-Bianchi and G. Lugosi. *Prediction, learning, and games*. Cambridge Univ Pr, 2006.

[23] S. Bubeck and N. Cesa-Bianchi. Regret analysis of stochastic and nonstochastic multi-armed bandit problems. *Foundations and Trends in Machine Learning*, 5(1):1–122, 2012.

[24] M. M. Drugan and A. Nowé. Designing multi-objective multi-armed bandits algorithms: A study. In *IJCNN*, pages 1–8, 2013.

[25] Yossi Azar, Uriel Feige, Michal Feldman, and Moshe Tennenholtz. Sequential decision making with vector outcomes. In *ITCS*, pages 195–206, 2014.

A Preference-Based Bandit Framework for Personalized Recommendation

Maryam Tavakol[1] and **Ulf Brefeld**[2]

Abstract. We present contextual bandits for personalized recommendation scenarios where user preferences are available. The model is a composite of a personalized model and a one-serves-all component where the latter resembles the mainstream recommendation. The derivation is consequentially carried out using Fenchel-Legrende conjugates and thus applicable in many different learning tasks. We present a unified framework that allows for quickly adapting our contextual bandits to different applications.

1 Introduction

Recommender systems are designed to serve user needs by extracting relevant content from a large amount of available information. User needs are generally characterized by individual interests of a user. However, considering many users at once gives rise to joint interests (e.g., topselling items) and needs that could be captured by a one-serves-all recommendation. Therefore, personalized recommendation aims to derive the individual preferences as well as their collective aggregate over all the users.

Traditional recommender systems focus on the recommendation problem from different perspectives. There are non-personalized approaches that focus on short-term goals and inferring topics of user sessions [7, 6], while collaborative filtering methods, on the other hand, aim to capture long-term preferences of users [4, 3]. The collaborative approach computes probably interesting items to a user by focusing on interests of similar peers. Whenever user preferences are available, it is convenient to directly learn user profiles from the partial order of items. Since preferences are often contradictive across serveral users, such an approach naturally extends to personalized recommendations with a dedicated model for every user. To also provide recommendations for new users with only little historical traffic, the individual models can be used as offsets to a one-serves-all (or average) model.

In this paper, we present a unified contextual bandit framework for personalized recommendation. The underlying scheme models the preferences between items which consists of an average part and an individual mnodel to compute the expected reward. We propose to leverage ideas from [5] to model our preference-based approach as a contextual bandit that is augmented by an individual offset. All derivations are carried out in dual space and using Fenchel-Legendre conjugates of the loss functions which renders our approach for a wide range of loss functions. In the next section, we derive a generalized model for personalized preference-based recommendation in dual space.

[1] Leuphana University of Lüneburg, email: tavakol@leuphana.de
[2] Leuphana University of Lüneburg, email: brefeld@leuphana.de

2 Linear Bandits in Dual Space

In this paper, we focus on sequential recommender systems for m users, $U = \{u_1, u_2, ..., u_m\}$, and n items, $A = \{a_1, a_2, ..., a_n\}$. Every item a_i is characterized by a set of attributes given by a feature vector $\boldsymbol{z}_i \in \mathbb{R}^k$. At each time step t, the goal of the system is to recommend items to the current user that are more likely to be clicked. We show how to derive the general optimization framework for linear bandits in dual space considering both the average and personalized models.

The proposed model is defined by a single bandit which learns the preferences between items for all the users. Assume that $\mathbf{z}_i$ and $\mathbf{z}_k$ belong to the items a_i and a_k, respectively, thus, we assign $\mathbf{z}_{i \succ k} := \mathbf{z}_i - \mathbf{z}_k$ to show the preference of item a_i over a_k. The payoff is therefore determined as a linear function of the preference,

$$\mathbb{E}[r_{t,i \succ k} | u_t = u_j] = \boldsymbol{\theta}^\top \mathbf{z}_{i \succ k} + \boldsymbol{\beta}_t^\top \mathbf{z}_{i \succ k} + b_{ik},$$

where $\boldsymbol{\theta}$ is the weight vector for the average model, while $\boldsymbol{\beta}_t = \boldsymbol{\beta}_j$ is the individual parameter for user j. $r_{t,i \succ k}$ shows the reward obtained by choosing item a_i over a_k at time t. For simplicity, we augment the feature vectors by a constant term (e.g., $\mathbf{z}_{i \succ k,0} = 1$) and move b_{ik} accordingly into the $\boldsymbol{\theta}$ and $\boldsymbol{\beta}$.

Let $a_t = a_i$ be the selected item to recommend at time t, the problem in [5] is then relaxed to a simple one-armed bandit which learns the preferences between contexts, i.e., item features,

$$h_{\boldsymbol{\theta}, \boldsymbol{\beta}_j}(\boldsymbol{z}_{i \succ l}; \exists l) = (\boldsymbol{\theta} + \boldsymbol{\beta}_j)^\top \boldsymbol{z}_{i \succ l},$$

where hypothesis h predicts the expected payoff for the specific user u_j on item a_i. Moreover, we substitute $\boldsymbol{z}_{i \succ l}$ by $\boldsymbol{z}_t$ as the context at time t. Given an appropriate loss function $V(., r_t)$, the regularized optimization problem can be stated as

$$\inf_{\substack{\boldsymbol{\theta} \\ \boldsymbol{\beta}_1, ..., \boldsymbol{\beta}_m}} \quad \frac{1}{T} \sum_{t=1}^{T} V([\boldsymbol{\theta} + \boldsymbol{\beta}_t]^\top \boldsymbol{z}_t, r_t) + \frac{\lambda}{2} \|\boldsymbol{\theta}\|^2 + \frac{\hat{\mu}}{2} \sum_j \|\boldsymbol{\beta}_j\|^2. \tag{1}$$

Let $C = \frac{1}{\lambda T}$ and $\mu = \frac{\bar{\mu}}{\lambda}$, by incorporating y_t as shorthand for the predicted payoff we have

$$\inf_{\substack{\boldsymbol{\theta}, \boldsymbol{y} \\ \boldsymbol{\beta}_1, ..., \boldsymbol{\beta}_m}} \quad C \sum_{t=1}^{T} V(y_t, r_t) + \frac{1}{2} \|\boldsymbol{\theta}\|^2 + \frac{\mu}{2} \sum_j \|\boldsymbol{\beta}_j\|^2$$
$$s.t. \quad \forall t: \quad (\boldsymbol{\theta} + \boldsymbol{\beta}_t)^\top \boldsymbol{z}_t = y_t,$$

The equivalent unconstrained problem is derived by incorporating

Lagrange multipliers, $\boldsymbol{\alpha} \in \mathbb{R}^T$,

$$\sup_{\boldsymbol{\alpha}} \inf_{\substack{\boldsymbol{\theta},\boldsymbol{y} \\ \boldsymbol{\beta}_1,\ldots,\boldsymbol{\beta}_m}} C\sum_{t=1}^{T} V(y_t, r_t) + \frac{1}{2}\|\boldsymbol{\theta}\|^2 + \frac{\mu}{2}\sum_j \|\boldsymbol{\beta}_j\|^2 - \sum_{t=1}^{T} \alpha_t([\boldsymbol{\theta} + \boldsymbol{\beta}_t]^\top \boldsymbol{z}_t - y_t).$$

Setting the partial derivatives w.r.t. $\boldsymbol{\theta}$ to zero, leads to $\boldsymbol{\theta} = \sum_{t=1}^{T} \alpha_t \boldsymbol{z}_t = Z^\top \boldsymbol{\alpha}$, where $Z \in \mathbb{R}^{T\times k}$ is the design matrix given by the training data. The derivatives w.r.t. $\boldsymbol{\beta}_j$ gives

$$\boldsymbol{\beta}_j = \frac{1}{\mu} \sum_{\substack{t \\ \boldsymbol{\beta}_t = \boldsymbol{\beta}_j}} \alpha_t \mathbf{z}_t = \frac{1}{\mu} \sum_t \phi_{jt} \alpha_t \mathbf{z}_t = \frac{1}{\mu}(Z \circ \boldsymbol{\phi_j})^\top \boldsymbol{\alpha},$$

where $\boldsymbol{\phi}_j \in \mathbb{R}^{T\times 1}$ is a binary vector which is 1 when $\boldsymbol{\beta}_t = \boldsymbol{\beta}_j$, and zero otherwise, and $\circ(.,.)$ stands for element-wise product. Substituting the optimality conditions into the optimization function yields

$$\sup_{\boldsymbol{\alpha}} \inf_{\boldsymbol{y}} C\sum_{t=1}^{T} [V(y_t, r_t) + \frac{1}{C}\alpha_t y_t] - \frac{1}{2}\boldsymbol{\alpha}^\top ZZ^\top \boldsymbol{\alpha} - \frac{1}{2\mu}\sum_j \boldsymbol{\alpha}^\top (Z \circ \boldsymbol{\phi_j})(Z \circ \boldsymbol{\phi_j})^\top \boldsymbol{\alpha}.$$

Recall that the Fenchel-Legendre conjugate of a function g is defined as $g^*(\boldsymbol{u}) = \sup_{\boldsymbol{x}} \boldsymbol{u}^\top \boldsymbol{x} - g(\boldsymbol{x})$ [2]. Thus, by moving the infimum inside the summation and given the dual loss

$$V^*(-\frac{\alpha_t}{C}, r_t) = \sup_{y_t} -\frac{\alpha_t}{C} y_t - V(y_t, r_t),$$

the generalized optimization problem in dual space reduces to

$$\sup_{\boldsymbol{\alpha}} -C\sum_{t=1}^{T} V^*(-\frac{\alpha_t}{C}, r_t) - \frac{1}{2}\boldsymbol{\alpha}^\top ZZ^\top \boldsymbol{\alpha} - \frac{1}{2\mu}\sum_j \boldsymbol{\alpha}^\top (Z \circ \boldsymbol{\phi_j})(Z \circ \boldsymbol{\phi_j})^\top \boldsymbol{\alpha}. \quad (2)$$

2.1 Upper Confidence Bound

The challenge in bandit-based approaches is to balance exploration and exploitation to minimize the regret. Auer [1] demonstrates that confidence bounds provide useful means to balance the two oppositional strategies. The idea is to use the predicted reward together with its confidence interval to reflect the uncertainty of the model given the actual context.

In our contextual bandit, the expected payoff is approximated by a linear model with an (arbitrary) loss function. The uncertainty U of the obtained value for each arm is therefore proportional to the variance σ^2 of the expected payoff, $U = c\sigma$, where σ^2 is estimated from training points in neighboring contexts as well as the model parameters. The uncertainty is added as an upper bound to the prediction to produce a confidence bound for selection strategy across the arms.

2.2 Optimization

Equation (2) can be optimized with standard techniques such as gradient-based approaches. The unconstrained problem needs to be maximized w.r.t. the dual parameters $\boldsymbol{\alpha}$ and is given by

$$\sup_{\boldsymbol{\alpha}} -C\mathbb{I}^\top V^*(-\frac{\boldsymbol{\alpha}}{C}, \boldsymbol{r}) - \frac{1}{2}\boldsymbol{\alpha}^\top ZZ^\top \boldsymbol{\alpha} - \frac{1}{2\mu}\sum_j \boldsymbol{\alpha}^\top (Z \circ \boldsymbol{\phi_j})(Z \circ \boldsymbol{\phi_j})^\top \boldsymbol{\alpha}.$$

The gradient wrt $\boldsymbol{\alpha}$ is obtained by setting the derivative to zero.

$$-C\frac{\partial V^*(-\frac{\boldsymbol{\alpha}}{C}, \boldsymbol{r})}{\partial \boldsymbol{\alpha}} - (ZZ^\top - \frac{1}{\mu}[\sum_j (Z \circ \boldsymbol{\phi_j})(Z \circ \boldsymbol{\phi_j})^\top])\boldsymbol{\alpha} = 0$$

The actual form of the gradient depends on the dual loss V^*. Note that instantiations often give rise to more efficient optimization techniques than the general form in Equation (2) allows. Nevertheless, the sketched gradient-based approach will always work in case a general optimiser is needed, e.g., in cases where several loss functions should be tried out. Once the optimal $\boldsymbol{\alpha}^{opt}$ has been found, it can be used to compute the primal parameters. Alternatively, a kernel $K_Z = \phi_Z(Z, Z)$ could be deployed in the dual representation to allow for non-linear and convoluted feature space.

2.2.1 *Instantiation: Squared Loss*

We present the optimization algorithm for the special case of squared loss. The dual of squared loss is given by

$$V^*(-\frac{\alpha_t}{C}, r_t) = \frac{1}{2C^2}\alpha_t^2 - \frac{1}{C}\alpha_t r_t,$$

which leads to the following objective,

$$\max_{\boldsymbol{\alpha}} -\frac{1}{2C}\boldsymbol{\alpha}^\top \boldsymbol{\alpha} + \boldsymbol{r}^\top \boldsymbol{\alpha} - \frac{1}{2}\boldsymbol{\alpha}^\top [ZZ^\top + \frac{1}{\mu}(\sum_i \boldsymbol{\phi}_i \otimes \boldsymbol{\phi}_i^\top) \circ ZZ^\top]\boldsymbol{\alpha},$$

where $\otimes$ denotes the vector outer product. Rephrasing the problem as a minimization task and setting $P = \frac{1}{C}\mathbb{I} + ZZ^\top + \frac{1}{\mu}(\sum_i \boldsymbol{\phi}_i \otimes \boldsymbol{\phi}_i^\top) \circ ZZ^\top$, and $\mathbf{q} = -\mathbf{r}$, the task becomes a standard quadratic optimization problem,

$$\min_{\boldsymbol{\alpha}} \frac{1}{2}\boldsymbol{\alpha}^\top P\boldsymbol{\alpha} + \mathbf{q}^\top \boldsymbol{\alpha}.$$

The confidence bound for the linear bandit setting with squared loss is given by

$$U = c\sqrt{\boldsymbol{z}_t^\top (Z^\top Z + \lambda I)^{-1} \boldsymbol{z}_t}.$$

REFERENCES

[1] Peter Auer, 'Using confidence bounds for exploitation-exploration tradeoffs', *Journal of Machine Learning Research*, **3**, 397–422, (2003).

[2] Stephen Boyd and Lieven Vandenberghe, *Convex optimization*, Cambridge University Press, 2004.

[3] Yifan Hu, Yehuda Koren, and Chris Volinsky, 'Collaborative filtering for implicit feedback datasets', in *Proceedings of the 8th IEEE International Conference on Data Mining*, pp. 263–272. IEEE, (2008).

[4] Yehuda Koren, Robert Bell, and Chris Volinsky, 'Matrix factorization techniques for recommender systems', *Computer*, **8**, 30–37, (2009).

[5] L. Li, W. Chu, J. Langford, and R. E. Schapire, 'A contextual-bandit approach to personalized news article recommendation', in *Proceedings of the International World Wide Web Conference*, (2010).

[6] Maryam Tavakol and Ulf Brefeld, 'Factored mdps for detecting topics of user sessions', in *Proceedings of the 8th ACM Conference on Recommender Systems*, pp. 33–40. ACM, (2014).

[7] Chong Wang and David M Blei, 'Collaborative topic modeling for recommending scientific articles', in *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 448–456. ACM, (2011).

Statistical Inference for Incomplete Ranking Data: A Comparison of two Likelihood-Based Estimators

Inés Couso[1] and **Mohsen Ahmadi**[2] and **Eyke Hüllermeier**[3]

Abstract. We consider the problem of statistical inference for ranking data, namely the problem of estimating a probability distribution on the permutation space. Since observed rankings could be incomplete in the sense of not comprising all choice alternatives, we propose to tackle the problem as one of learning from imprecise or coarse data. To this end, we associate an incomplete ranking with its set of consistent completions. We instantiate and compare two likelihood-based approaches that have been proposed in the literature for learning from set-valued data, the marginal and the so-called face-value likelihood. Concretely, we analyze a setting in which the underlying distribution is Plackett-Luce and observations are given in the form of pairwise comparisons.

1 INTRODUCTION

The study of rank data and related probabilistic models on the permutation space (symmetric group) has a long tradition in statistics, and corresponding methods for parameter estimation have been used in various fields of application, such as psychology and the social sciences [9]. More recently, applications in information retrieval (search engines) and machine learning (personalization, preference learning) have caused a renewed interest in the analysis of rankings and topics such as "learning-to-rank" [8]. Indeed, methods for learning and constructing preference models from explicit or implicit preference information and feedback are among the recent research trends in these disciplines [5].

In most applications, the rankings observed are *incomplete* or *partial* in the sense of including only a subset of the underlying choice alternatives, whereas no preferences are revealed about the remaining ones—pairwise comparisons can be seen as an important special case. In this paper, we therefore approach the problem of learning from ranking data from the point of view of statistical inference with *imprecise data*. The key idea is to consider an incomplete ranking as a *set-valued observation*, namely the set of all complete rankings consistent with the incomplete observation. This approach is especially motivated by recent work on learning from imprecise, incomplete or fuzzy data [11, 3, 4, 6, 10].

Thus, our paper can be seen as an application of general methods proposed in that field to the specific case of ranking data. This is arguably interesting for both sides, research on statistics with imprecise data and learning from ranking data: For the former, ranking data is an interesting test bed that may help understand, analyze and compare methods for learning from imprecise data; for the latter, general approaches for learning from imprecise data may turn into new statistical methods for ranking.

In this paper, we compare two likelihood-based approaches for learning from imprecise data. More specifically, both approaches are used for inference about the so-called Plackett-Luce model, a parametric family of probability distributions on the permutation space.

2 PRELIMINARIES AND NOTATION

Let $\mathbb{S}_K$ denote the collection of rankings (permutations) over a set $U = \{a_1, \ldots, a_k\}$ of K items a_k, $k \in [K] = \{1, \ldots, K\}$. We denote by $\pi : [K] \longrightarrow [K]$ a complete ranking (a generic element of $\mathbb{S}_K$), where $\pi(k)$ denotes the position of the k^{th} item a_k in the ranking, and by π^{-1} the ordering associated with a ranking, i.e., $\pi^{-1}(j)$ is the index of the item on position j. We write rankings in brackets and orderings in parentheses; for example, $\pi = [2, 4, 3, 1]$ and $\pi^{-1} = (4, 1, 3, 2)$ both denote the ranking $a_4 \succ a_1 \succ a_3 \succ a_2$.

For a possibly incomplete ranking, which includes only some of the items, we use the symbol τ (instead of π). If the k^{th} item does not occur in a ranking, then $\tau(k) = 0$ by definition; otherwise, $\tau(k)$ is the rank of the k^{th} item. In the corresponding ordering, the missing items do simply not occur. For example, the ranking $a_4 \succ a_1 \succ a_2$ would be encoded as $\tau = [2, 3, 0, 1]$ and $\tau^{-1} = (4, 1, 2)$, respectively. We let $I(\tau) = \{k : \tau(k) > 0\} \subset [K]$ and denote the set of all rankings (complete or incomplete) by $\overline{\mathbb{S}}_K$.

An incomplete ranking τ can be associated with its set of linear extensions $E(\tau) \subset \mathbb{S}_K$, where

$$E(\tau) = \Big\{\pi : (\tau(i) - \tau(j))(\pi(i) - \pi(j)) \geq 0 \text{ for all } i, j \in I(\tau)\Big\}$$

An important special case is an incomplete ranking τ in the form of a pairwise comparison $a_i \succ a_j$ (i.e., $\tau(i) = 1$, $\tau(j) = 2$, $\tau(k) = 0$ otherwise), which is associated with the set of extensions

$$E(\tau) = E(a_i \succ a_j) = \{\pi \in \mathbb{S}_K : \pi(i) < \pi(j)\} \ .$$

Modeling an incomplete observation τ by the set of linear extensions $E(\tau)$ reflects the idea that τ has been produced from an underlying complete ranking π by some "coarsening" or "imprecisiation" process, which essentially consists of omitting some of the items from the ranking. $E(\tau)$ then corresponds to the set of all possible candidates π, i.e., all complete rankings that are compatible with the observation τ if nothing is known about the coarsening, except that it does not change the relative order of any items.

Sometimes, more knowledge about the coarsening is available, or reasonable assumptions can be made. For example, it might be known that τ is a top-t ranking, which means that it consists of the

[1] University of Oviedo, Spain, email: couso@uniovi.es
[2] Paderborn University, Germany, email:ahmadim@mail.upb.de
[3] Paderborn University, Germany, email:eyke@upb.de

items that occupy the first t positions in π. We use the following notation:

$$T(a_{i_1} \ldots, a_{i_t}) = \left\{ \pi \in \mathbb{S}_K \,:\, \pi(i_1) = 1, \ldots, \pi(i_t) = t \right\} .$$

Sometimes, we simply write $T(i_1, \ldots, i_t)$ instead of $T(a_{i_1}, \ldots, a_{i_t})$. Note that top-t information is more specific than simply observing the corresponding ranking of length t, i.e., $T(a_{i_1}, \ldots, a_{i_t}) \subset E(\tau)$ for $\tau^{-1} = (i_1, \ldots, i_t)$. Moreover, note that the collection of top-t rankings forms a partition of $\mathbb{S}_K$, i.e., $T(i_1, \ldots, i_t) \cap T(j_1, \ldots, j_t) = \emptyset$ for $(i_1, \ldots, i_t) \neq (j_1, \ldots, j_t)$.

3 PROBABILISTIC MODELS

Statistical inference requires a probabilistic model of the underlying data generating process, which, in our case, essentially comes down to specifying a probability distribution on the permutation space. One of the most well-known probability distributions of that kind is the Plackett-Luce (PL) model [9].

3.1 The Plackett-Luce Model

The PL model is parameterized by a vector $\theta = (\theta_1, \theta_2, \ldots, \theta_K) \in \Theta = \mathbb{R}_+^K$. Each θ_i can be interpreted as the weight or "strength" of the option a_i. The probability assigned by the PL model to a ranking represented by a permutation $\pi \in \mathbb{S}_K$ is given by

$$\mathrm{PL}_\theta(\{\pi\}) \;=\; \prod_{i=1}^{K} \frac{\theta_{\pi^{-1}(i)}}{\theta_{\pi^{-1}(i)} + \theta_{\pi^{-1}(i+1)} + \ldots + \theta_{\pi^{-1}(K)}} \qquad (1)$$

Obviously, the PL model is invariant toward multiplication of θ with a constant $c > 0$, i.e., $\mathrm{PL}_\theta(\pi) = \mathrm{PL}_{c\theta}(\pi)$ for all $\pi \in \mathbb{S}_K$ and $c > 0$. Consequently, θ can be normalized without loss of generality (and the number of degrees of freedom is only $K - 1$ instead of K). Note that the most probable ranking, i.e., the mode of the PL distribution, is simply obtained by sorting the items in decreasing order of their weight:

$$\pi^* = \operatorname*{arg\,max}_{\pi \in \mathbb{S}_K} \mathrm{PL}_\theta(\pi) = \operatorname*{arg\,sort}_{k \in [K]} \{\theta_1, \ldots, \theta_K\} . \qquad (2)$$

As a convenient property of PL, let us mention that it allows for a very easy computation of marginals, because the marginal probability on a subset $U' = \{a_{i_1}, \ldots, a_{i_J}\} \subset U$ of $J \leq K$ items is again a PL model parametrized by $(\theta_{i_1}, \ldots, \theta_{i_J})$. Thus, for every $\tau \in \overline{\mathbb{S}}_K$ with $I(\tau) = U'$,

$$\mathrm{PL}_\theta(E(\tau)) \;=\; \prod_{j=1}^{J} \frac{\theta_{\tau^{-1}(j)}}{\theta_{\tau^{-1}(j)} + \theta_{\tau^{-1}(j+1)} + \ldots + \theta_{\tau^{-1}(J)}} \qquad (3)$$

In particular, this yields pairwise probabilities

$$\mathrm{PL}_\theta(E(a_i \succ a_j)) = \frac{\theta_i}{\theta_i + \theta_j} .$$

This is the well-known Bradley-Terry model [9], a model for the pairwise comparison of alternatives. Obviously, the larger θ_i in comparison to θ_j, the higher the probability that a_i is chosen. The PL model can be seen as an extension of this principle to more than two items: the larger the parameter θ_i in (1) in comparison to the parameters θ_j, $j \neq i$, the higher the probability that a_i appears on a top rank.

3.2 A Stochastic Model of Coarsening

While (1) defines a probability for every *complete* ranking π, and hence a distribution $P : \mathbb{S}_K \longrightarrow [0, 1]$, an extension of P from $\mathbb{S}_K$ to $\overline{\mathbb{S}}_K$ is in principle offered by (3). One should note, however, that marginalization in the traditional sense is different from coarsening. In fact, (3) assumes the subset of items U' to be fixed beforehand, prior to drawing a ranking at random. For example, focusing on two items a_i and a_j, one may ask for the probability that a_i will precede a_j in the next ranking drawn at random according to P.

Recalling our idea of a coarsening process, it is more natural to consider the data generating process as a two step procedure:

$$p_{\theta,\lambda}(\tau, \pi) = p_\theta(\pi) \cdot p_\lambda(\tau \,|\, \pi) \qquad (4)$$

According to this model, a complete ranking π is generated first according to $P_\theta(\cdot)$, and this ranking is then turned into an incomplete ranking τ according to $P_\lambda(\cdot \,|\, \pi)$. Thus, the coarsening process is specified by a family of conditional probability distributions

$$\left\{ P_\lambda(\cdot \,|\, \pi) \,:\, \pi \in \mathbb{S}_K,\, \lambda \in \Lambda \right\} , \qquad (5)$$

where λ collects all parameters of these distributions. $P_{\theta,\lambda}(\tau, \pi)$ is the probability of producing the data $(\tau, \pi) \in \overline{\mathbb{S}}_K \times \mathbb{S}_K$. Note, however, that π is actually not observed.

4 STATISTICAL INFERENCE

As for the statistical inference about the process (4), our main interest concerns the "precise part", i.e., the parameter θ, whereas the coarsening is rather considered as a complication of the estimation. In other words, we are less interested in inference about λ or, stated differently, we are interested in λ only in so far as it helps to estimate θ. In this regard, it should also be noted that inference about λ will generally be difficult: Due to the sheer size of the family of distributions (5), λ could be very high-dimensional. Besides, concrete model assumptions about the coarsening process may not be obvious.

Therefore, what we are mainly aiming for is an estimation technique that is efficient in the sense of circumventing direct inference about λ, and at the same time robust in the sense that it yields reasonably good results for a wide range of coarsening procedures, i.e., under very weak assumptions about the coarsening (or perhaps no assumptions at all). As a first step toward this goal, we look at two estimation principles that have recently been proposed in the literature, both being based on the principle of likelihood maximization.

In the following, the random variable X will denote the precise outcome of a single random experiment, i.e., a complete ranking π, whereas Y denotes the coarsening τ. We assume to be given an i.i.d. sample of size N and let $\boldsymbol{\tau} = (\tau_1, \ldots, \tau_N) \in (\overline{\mathbb{S}}_K)^N$ denote a sequence of N independent incomplete observations of Y.

4.1 The Marginal Likelihood

The perhaps most natural approach is to consider the marginal likelihood function (also called "visible likelihood" in [1]), i.e., the probability of the observed data Y given the parameters θ and λ:

$$\begin{aligned} L_V(\theta, \lambda) = P(\boldsymbol{\tau} \,|\, \theta, \lambda) &= \prod_{i=1}^{N} P(Y = \tau_i \,|\, \theta, \lambda) \qquad (6) \\ &= \prod_{i=1}^{N} \sum_{\pi \in \mathbb{S}_K} p_\theta(\pi) p_\lambda(\tau_i \,|\, \pi) \end{aligned}$$

The maximum likelihood estimate (MLE) would then be given by

$$(\theta^*, \lambda^*) = \operatorname*{arg\,max}_{(\theta,\lambda)\in\Theta\times\Lambda} L_V(\theta, \lambda)\,,$$

or, emphasizing inference about θ, by

$$\theta^* = \operatorname*{arg\,max}_{\theta\in\Theta} \max_{\lambda\in\Lambda} L_V(\theta, \lambda)\,.$$

As can be seen, this approach requires assumptions about the parametrization of the coarsening, i.e., the parameter space Λ. Of course, since both $\mathbb{S}_K$ and $\overline{\mathbb{S}}_K$ are finite, these assumptions can be "vacuous" in the sense of allowing all possible distributions. Thus, the family (5) would be specified in a tabular form by letting

$$p_\lambda(\tau \,|\, \pi) = \lambda_{\pi,\tau} \tag{7}$$

for all $\tau \in \overline{\mathbb{S}}_K$ and $\pi \in E(\tau)$ (recall that $P(\tau \,|\, \pi) = 0$ for $\pi \notin E(\tau)$). In other words, Λ is given by the set of all these parametrizations under the constraint that

$$\sum_{\tau\in\overline{\mathbb{S}}_K} \lambda_{\pi,\tau} = 1$$

for all $\pi \in \mathbb{S}_K$. We denote this parametrization by Λ_{vac}.

4.2 The Face-Value Likelihood

The *face-value likelihood* is expressed as follows [2, 7]:

$$\begin{aligned} L_F(\theta, \lambda) &= \prod_{i=1}^{N} P\big(X \in E(\tau_i) \,|\, \theta, \lambda\big) \\ &= \prod_{i=1}^{N} \sum_{\pi\in E(\tau_i)} p_\theta(\pi) \end{aligned} \tag{8}$$

Note that the face-value likelihood does actually not depend on λ, which means that we could in principle write $L_F(\theta)$ instead of $L_F(\theta, \lambda)$. Indeed, this approach does not explicitly account for the coarsening process, or at least does not consider the coarsening as a stochastic process. The only way of incorporating knowledge about this process is to replace the set of linear extensions, $E(\tau_i)$, with a smaller set of complete rankings associated with an incomplete observation τ_i. This can be done if the coarsening is deterministic, like in the case of top-t selection.

5 COMPARISON OF THE APPROACHES

These two likelihood functions (6) and (8) coincide when the collection of possible values for Y forms a partition of the collection of permutations $\mathbb{S}_K$, since the events $Y = \tau_i$ and $X \in E(\tau_i)$ are then the same. But they do not coincide in the general case, where the event $Y = \tau_i$ implies but does not necessarily coincide with $X \in E(\tau_i)$.

In the following, we refer to the parameter estimation via maximization of (6) and (8) as MLM (marginal likelihood maximization) and FLM (face-value likelihood maximization), respectively.

5.1 Known Coarsening

A comparison between the marginal and face-value likelihood is arguable in the case where the coarsening is assumed to be known, because, as already said, the face-value likelihood is not able to exploit this knowledge (unless the coarsening is deterministic and forms a partition). Obviously, ignorance of the coarsening may lead to very poor estimates in general, as shown by the following example.

Let $K = 3$ and $U = \{a_1, a_2, a_3\}$. To simplify notation, we denote a ranking $a_i \succ a_j \succ a_k$ by $a_ia_ja_k$. We assume the PL model and suppose the coarsening to be specified by the following (deterministic) relation between complete rankings π and incomplete observations τ, which are all given in the form of pairwise comparisons:

	a_1a_2	a_2a_1	a_1a_3	a_3a_1	a_2a_3	a_3a_2
$a_1a_2a_3$	0	0	0	0	1	0
$a_1a_3a_2$	0	0	0	0	0	1
$a_2a_1a_3$	0	1	0	0	0	0
$a_2a_3a_1$	0	1	0	0	0	0
$a_3a_1a_2$	0	0	0	1	0	0
$a_3a_2a_1$	0	1	0	0	0	0

Denoting by n_{ij} the number of times $a_i \succ a_j$ has been observed, the face-value likelihood function reads as follows:

$$L_F(\boldsymbol{\tau}; \theta) = \prod_{i=1}^{3} \prod_{j\neq i} \left(\frac{\theta_i}{\theta_i + \theta_j}\right)^{n_{ij}}\,.$$

According to the above relation, we have the following:

$$\begin{aligned} n_{12} &= 0 \\ n_{21} &= n_{213} + n_{231} + n_{321} \\ n_{12} &= 0 \\ n_{13} &= n_{312} \\ n_{23} &= n_{123} \\ n_{32} &= n_{132} \end{aligned}$$

Therefore,

$$\begin{aligned} L_F(\theta) = &\left(\frac{\theta_2}{\theta_1 + \theta_2}\right)^{n_{213}+n_{231}+n_{321}} \\ &\times \left(\frac{\theta_3}{\theta_1 + \theta_3}\right)^{n_{312}} \times \left(\frac{\theta_2}{\theta_2 + \theta_3}\right)^{n_{123}} \times \left(\frac{\theta_3}{\theta_2 + \theta_3}\right)^{n_{132}}\,. \end{aligned}$$

For an arbitrary triplet $\theta = (\theta_1, \theta_2, \theta_2)$ with $\theta_1 + \theta_2 + \theta_3 = 1$, we observe that

$$L_F(\theta_1, \theta_2, \theta_3) \leq L_F\left(0, \theta_2', \theta_3'\right)\,,$$

where $\theta_2' = \frac{\theta_2}{\theta_2+\theta_3}$ and $\theta_3' = \frac{\theta_3}{\theta_2+\theta_3}$. In fact,

$$L_F(0, \theta_2', \theta_3') = (\theta_2')^{n_{123}} \cdot (\theta_3')^{n_{132}}\,,$$

and therefore

$$\begin{aligned} L_F(\theta) = &\left[\left(\frac{\theta_2}{\theta_1 + \theta_2}\right)^{n_{213}+n_{231}+n_{321}} \cdot \left(\frac{\theta_3}{\theta_1 + \theta_3}\right)^{n_{312}}\right] \\ &\times L_F(0, \theta_2', \theta_3')\,, \end{aligned}$$

which is clearly less than or equal to $L_F(0, \theta_2', \theta_3')$. Furthermore, according to Gibb's inequality, the above likelihood value, $L_F(0, \theta_2', \theta_3')$, is maximized for

$$(\hat{\theta}_2', \hat{\theta}_3') = \left(\frac{n_{123}}{n_{123} + n_{132}}, \frac{n_{132}}{n_{123} + n_{132}}\right)\,.$$

Thus, if we for instance assume that the true distribution over $\mathbb{S}_3$ is PL with parameter $\theta = (\theta_1, \theta_2, \theta_3) = (0.99, 0.005, 0.005)$, then our estimation of θ based on the face-value likelihood function will be

$$\left(0, \frac{n_{123}}{n_{123} + n_{132}}, \frac{n_{132}}{n_{123} + n_{132}}\right)\,,$$

which tends to $(0, 0.5, 0.5)$ as n tends to infinity.

5.2 Unknown Coarsening

The comparison between the two approaches appears to be more reasonable when the coarsening is assumed to be unknown. In that case, it might be fair to instantiate the marginal likelihood with the parametrization Λ_{vac}, because just like the face-value likelihood, it is then essentially ignorant about the coarsening. However, the estimation of the coarsening process under Λ_{vac} is in general not practicable, simply because the number of parameters (7) is too large: One parameter $\lambda_{\pi,\tau}$ for each $\tau \in \overline{\mathbb{S}}_K$ and $\pi \in E(\tau)$ makes about $2^K K!$ parameters in total. Besides, Λ_{vac} may cause problems of model identifiability. What we need, therefore, is a simplifying assumption on the coarsening.

5.2.1 Rank-Dependent Coarsening

The assumption we make here is a property we call *rank-dependent* coarsening. A coarsening procedure is rank-dependent if the incompletion is only acting on *ranks* (positions) but not on *items*. That is, the procedure randomly selects a subset of ranks and removes the items on these ranks, independently of the items themselves. In other words, an incomplete observation τ is obtained by projecting a complete ranking π on a random subset of positions $A \in 2^{[K]}$, i.e., the family (5) of distributions $P_\lambda(\cdot \,|\, \pi)$ is specified by a single measure on $2^{[K]}$. Or, stated more formally,

$$P\big(\pi^{-1}(A) \,|\, \pi^{-1}\big) = P\big(\sigma^{-1}(A) \,|\, \sigma^{-1}\big)$$

for all $\pi, \sigma \in \mathbb{S}^K$ and $A \subset [K]$, where $\pi^{-1}(A)$ denotes the projection of the ordering π^{-1} to the positions in A.

In the following, we make an even stronger assumption and assume obervations in the form of (rank-dependent) pairwise comparisons. In this case, the coarsening is specified by probabilities

$$\Big\{\lambda_{i,j} \,|\, 1 \leq i < j \leq K,\, \lambda_{i,j} \geq 0,\, \sum_{1 \leq i < j \leq K} \lambda_{i,j} = 1\Big\},$$

where $\lambda_{i,j}$ denotes the probability that the ranks i and j are selected.

5.2.2 Likelihoods

Under the assumption of the PL model and rank-dependent pairwise comparisons as observations, the marginal likelihood for an observed set of pairwise comparisons $a_{i_n} \succ a_{j_n}$, $n \in [N]$, is given by

$$L_V(\theta, \lambda) = \prod_{n=1}^{N} \sum_{\pi \in \mathbb{S}_K, \pi(i_n) < \pi(j_n)} \lambda_{\pi(i_n),\pi(j_n)}\, \mathrm{PL}_\theta(\pi)\,. \tag{9}$$

The corresponding expression for the face-value likelihood is

$$L_F(\theta) = \prod_{n=1}^{N} \frac{\theta_{i_n}}{\theta_{i_n} + \theta_{j_n}}\,. \tag{10}$$

Obviously, since the face-value likelihood is ignorant of the coarsening, we cannot expect the maximizer $\hat{\theta}$ of (10) to coincide with the true parameter θ. Interestingly, however, we can prove these parameters to be *comonotonic*, which is enough to recover the most probable ranking (2).

Theorem 1 *Suppose complete rankings to be generated by the PL model with parameters $\theta_1 > \theta_2 > \ldots > \theta_K$. Moreover, let the coarsening procedure be given by a rank-dependent selection of pairwise comparisons between items. Then, with an increasing sample size $N \to \infty$, the maximizer $\hat{\theta}$ of (10) satisfies*

$$\hat{\theta}_1 > \hat{\theta}_2 > \ldots > \hat{\theta}_K$$

with probability 1.

According to this result, we can expect the face-value likelihood to yield reasonably strong estimates

$$\hat{\pi} = \operatorname*{arg\,sort}_{k \in [K]} \left\{\hat{\theta}_1, \ldots, \hat{\theta}_K\right\}, \tag{11}$$

although the parameter $\hat{\theta}$ itself might be biased.

5.2.3 Experiments

In order to compare the two approaches experimentally, synthetic data was produced by fixing parameters θ and λ and drawing N samples at random according to (4). Then, estimations $\hat{\theta}$ and $\hat{\pi}$ were obtained for both likelihoods, i.e., by maximizing (9) and (10). As a baseline, we also included estimates of θ assuming the coarsening λ to be known; to this end, (9) is maximized as a function of θ only. The three approaches are called MLM, FLM, and TLM, respectively.

The quality of estimates is measured both for the parameters and the induced rankings (11): in terms of the Euclidean distance between θ and $\hat{\theta}$, and in terms of the Kendall distance (relative number of pairwise inversions between items) between π and $\hat{\pi}$. The expectations of the quality measures were approximated by averaging over 100 simulation runs.

Here, we present results for a series of experiments with $K = 4$, $\theta = (0.4, 0.3, 0.2, 0.1)$, and different assumptions on the coarsening:

- In the first experiment, we set $\lambda_{1,2} = \ldots = \lambda_{3,4} = 1/6$. Thus, pairwise comparisons are selected uniformly at random. In this case, the face-value likelihood coincides with the likelihood of θ assuming the coarsening to be known, so this setting is clearly in favor of FLM (which, as already said, also coincides with TLM). Indeed, as can be seen in Figure 1 (top), FLM yields very accurate estimates that improve with an increasing sample size. Nevertheless, MLM is not much worse and performs more or less on a par.
- In the second experiment, $\lambda_{1,2} = 1$ and $\lambda_{1,3} = \lambda_{1,4} = \ldots = \lambda_{3,4} = 0$. This corresponds to the top-2 setting, in which always the two items on the top of the ranking are observed. As expected, FLM now performs worse than MLM. As can be seen in Figure 1 (middle, left), the parameter estimates of FLM are biased. Nevertheless, the estimation $\hat{\pi}$ is still decent (Figure 1, middle, right) and continues to improve with increasing sample size; this is in agreement with Theorem 1, according to which the Kendall distance will tend to 0 for $N \to \infty$.
- In the last experiment, items are selected with a probability inversely proportional their ranks: $\lambda_{i,j} \propto (8 - i - j)$. Thus, pairs on better ranks are selected with a higher probability than pairs on lower ranks. The results are shown in Figure 1 (bottom). As can be seen, FLM is again biased and performs worse than MLM. However, the bias and the difference in performance are much smaller than in the top-2 scenario. This is hardly surprising, given that the coarsening λ in this experiment is less extreme than in the top-2 case. Instead, it is closer to the uniform coarsening of the first experiment, which, as already said, is the coarsening that FLM is right for.

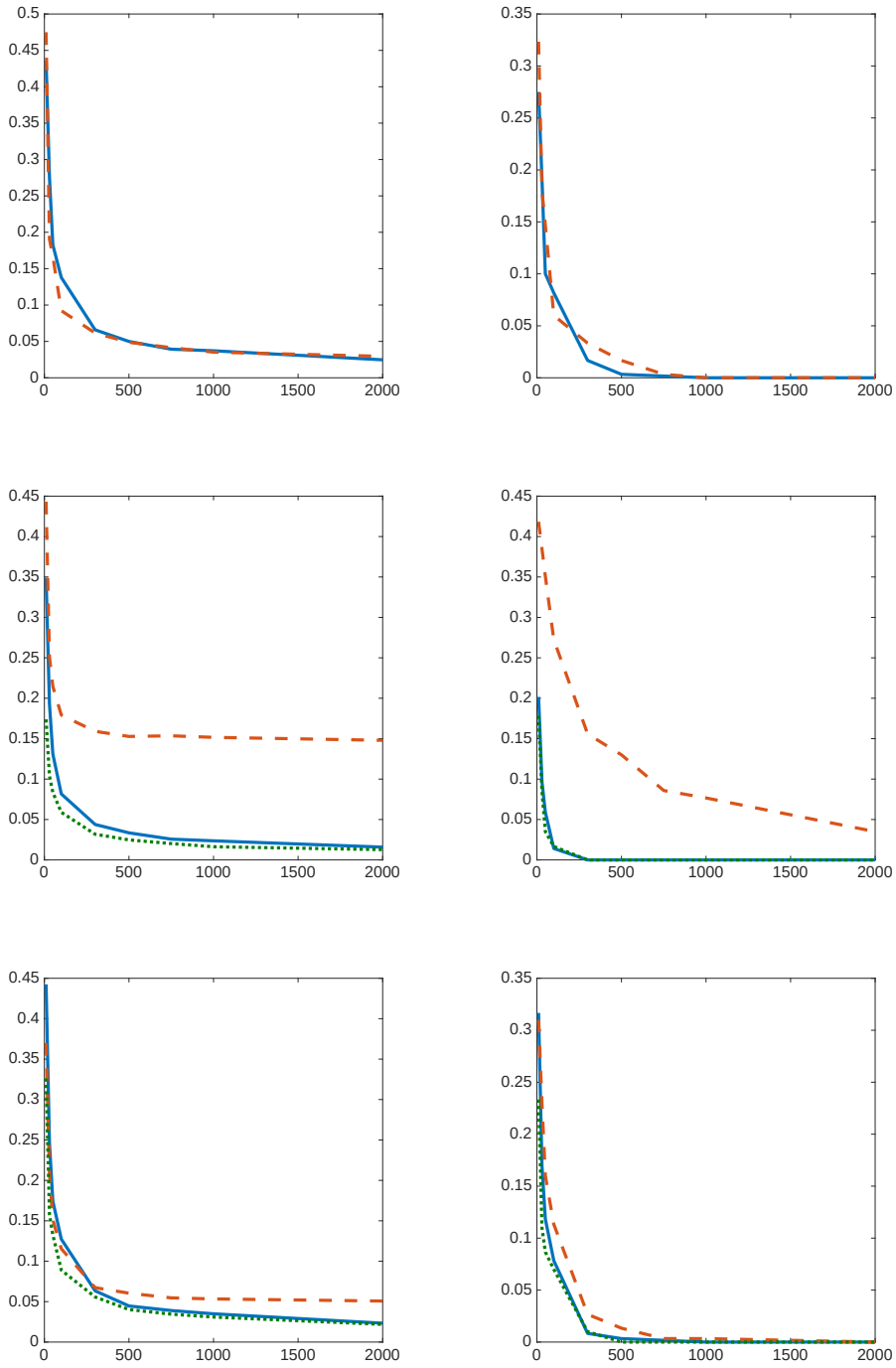


Figure 1. Euclidean distance of parameter estimate $\hat{\theta}$ (left column) and Kendall distance of predicted ranking $\hat{\pi}$ (right column) for three experimental settings: uniform selection of pairwise comparisons (top), top-2 selection (middle), and rank-proportional selection (bottom). Curves are plotted in solid lines for MLM, dashed for FLM, and dotted for TLM.

6 CONCLUSION

This paper is meant as a first step toward learning from incomplete ranking data based on methods for learning from imprecise (set-valued) data. Needless to say, the scope of the paper is very limited, both in terms of the methods considered (inference based on the marginal and the face-value likelihood) and the setting analyzed (observation of pairwise comparisons based on the PL model with rank-dependent coarsening)—generalizations in both directions shall be considered in future work. Nevertheless, our results clearly reveal some important points:

- The arguably "correct" way of tackling the problem is complete inference about (θ, λ), i.e., about the complete data generating process, as done by MLM. While this approach will guarantee theoretically optimal results, it will not be practicable in general, unless the number of items is small or the parametrization of the coarsening process is simplified by very restrictive assumptions.
- Simplified estimation techniques such as FLM, which make incorrect assumptions about the coarsening or even ignore this process altogether, will generally lead to biased results.
- Yet, in the context of ranking data, one has to distinguish between the estimation of the parameter θ, i.e., the identification of the model, and the prediction of a related ranking π (typically the most probable ranking given θ, i.e., the mode of the distribution). Indeed, the main interest often concerns π, while θ only serves an auxiliary purpose. As we have shown for FLM, both theoretically and experimentally, a biased estimation of θ does not exclude an accurate prediction of π, at least under certain assumptions on the coarsening process.

These observations suggest a natural direction for future work, namely the search for methods that achieve a reasonable compromise in the sense of being practicable and robust at the same time, where we consider a method robust if it guarantees a strong performance over a broad range of relevant coarsening procedures. Such methods should improve on techniques that ignore the coarsening, albeit at an acceptable increase in complexity.

REFERENCES

[1] I. Couso and D. Dubois, 'A general framework for maximizing likelihood under incomplete data'. Submitted for publication.

[2] A.P. Dawid and J.M. Dickey, 'Likelihood and bayesian inference from selectively reported data', *Journal of the American Statistical Association*, **72**, 845–850, (1977).

[3] T. Denoeux, 'Maximum likelihood estimation from fuzzy data using the EM algorithm', *Fuzzy Sets and Systems*, **183**(1), 72–91, (2011).

[4] T. Denoeux, 'Maximum likelihood estimation from uncertain data in the belief function framework', *IEEE Transactions on Knowledge and Data Engineering*, **25**(1), 119–130, (2013).

[5] J. Fürnkranz and E. Hüllermeier, *Preference Learning*, Springer-Verlag, 2010.

[6] E. Hüllermeier, 'Learning from imprecise and fuzzy observations: Data disambiguation through generalized loss minimization', *International Journal of Approximate Reasoning*, **55**(7), 1519–1534, (2014).

[7] M. Jaeger, 'Ignorability for categorical data', *The Annals of Statistics*, **33**(4), 1964–1981, (2005).

[8] T.Y. Liu, *Learning to Rank for Information Retrieval*, Springer-Verlag, 2011.

[9] J.I. Marden, *Analyzing and Modeling Rank Data*, London, New York, 1995.

[10] J. Plass, M. Cattaneo, G. Schollmeyer, and T. Augustin, 'Testing of coarsening mechanism: Coarsening at random versus subgroup independence', in *Proc. SMPS 2016, 8th Int. Conf. on Soft Methods in Probability and Statistics*, pp. 415–422. Springer-Verlag, (2016).

[11] R. Viertl, *Statistical Methods for Fuzzy Data*, Wiley, 2011.

A Noisy Sorting-based Ranking Model

Adil Paul[1] and **Róbert Busa-Fekete**[1]

Abstract. In this paper, we study a ranking model based on noisy sorting; the process of ordering a set of elements based on the outcomes of stochastic pairwise comparisons between them. Our ranking model falls into the class of log-linear models where the parameters are associated with the pairs of objects involved in the sorting process. We show that this model can be written in a closed form for insertion sort and the parameter estimation can be carried out based on Generalized Iterative Scaling. Furthermore, we make use of the Metropolis-Hastings algorithm with Mallows model as proposal distribution to sample our model.

1 Introduction

Noisy sorting consists of sorting a set of elements $\mathcal{O} = \{o_1, o_2, \ldots, o_K\}$ based on stochastic pairwise comparisons. More detailed, we assume that the outcome when comparing two elements, say $o_i, o_j \in \mathcal{O}$, is stochastic so as the probability of an element o_i is preferred to o_j when these two elements are compared to each other is $p_{i,j} = \mathbb{P}(o_i \succ o_j)$. Then any sorting algorithm that operates on stochastic pairwise comparisons, outputs a random ordering over the set of elements that depends on the stochastic pairwise comparisons involved in the sorting algorithm. In this way, one can define a distribution over the set of orderings of the set $\mathcal{O}$ based on noisy sorting.

Ranking models based on noisy sorting are used in statistical analysis of ranking data [6, 1, 7]. However, fitting these models to real world data is often a challenging task. In this paper, we propose a noisy sorting-based ranking model which is a compound model, that is, the probability of observing an ordering is computed as the product of probabilities with respect to noisy sorting models. We shall focus on insertion sort with non-stationary stochastic pairwise comparisons, i.e. the matrix of pairwise probabilities $\mathbf{P} = [p_{ij}]_{1 \leq i,j \leq K}$ can be arbitrary up to the pairwise reciprocal constraint $p_{ij} = 1 - p_{ji}$. We show that the parameter estimation can be carried out based on Generalized Iterative Scaling [3] thanks to the fact that the model can be written in a closed form as a log-linear model. In addition we make use of the Metropolis-Hastings algorithm with Mallows model as proposal distribution to sample the model.

2 Ranking distribution based on noisy sorting model

A sorting algorithm puts the objects stored in a list in a certain order based on pairwise comparisons of the objects. Most often, these objects to be sorted are numbers, and the pairwise comparison is determined based on some binary relation, for example, $\leq$ relation for increasing order and $\geq$ relation for decreasing order. Note that the input list of the sorting algorithm does not affect its output, but might do for its time complexity. Therefore, the time complexity of the sorting algorithms is often computed by assuming uniform distribution over the possible inputs. This kind of analysis is known as average time complexity analysis.

Any pairwise comparison-based sorting algorithm can be extended to a noisy sorting model by using stochastic pairwise comparisons. More detailed, when a sorting algorithm requires the comparison of two objects, say o_i and o_j, a coin with success probability p_{ij} is flipped and the outcome determines the order of the two elements, i.e. o_i is preferred to o_j if the outcome of the coin-flipping is 1, and o_j is preferred to o_i otherwise. We furthermore assume that the relation $p_{i,j}$ is reciprocal in a sense that $p_{i,j} = 1 - p_{j,i}$ for all $i, j \in [K]$. Note that the output of the noisy sorting model is a random ordering of the objects that depends on the success probabilities or *pairwise probabilities* $\mathbf{P} = [p_{i,j}]_{1 \leq i,j \leq K}$, and also on the input list of the objects which we call *initial ordering* of the objects. The following example elaborates on this dependence.

Example 1. *Consider the insertion sort algorithm with two different initial orderings $\pi = (o_1, o_2, o_3)$ and $\pi' = (o_3, o_2, o_1)$. Let the pairwise probabilities be $p_{1,2} = 1/4$, and $p_{1,3} = p_{2,3} = 1/2$. Now let us compute the probability of observing $\sigma = (o_1, o_2, o_3)$. According to the definition of insertion sort, when we start from π, first o_1 is taken, and then o_2 is inserted after o_1, and o_3 after o_2. Therefore the probability of observing σ starting from π is $p_{1,2}p_{1,3}p_{2,3} = 0.0625$. Whereas it is easy to see that the probability of observing σ starting from π' is $p_{1,2}p_{2,3} = 0.125$.*

Based on the noisy sorting model, one can define a distribution over the symmetric group $\mathbb{S}_K$ of order K which represents the set of all orderings of K objects. We denote the probability distribution over $\mathbb{S}_K$ induced by a noisy sorting model by $\mathbb{P}_{\mathcal{A}}(\sigma|\pi; P)$ where $\mathcal{A}$ is the sorting algorithm, π is the initial ordering, σ is an ordering and $\mathbf{P}$ is the matrix containing the pairwise probabilities. The initial ordering π is a latent variable of the model. The models we shall consider here can be written in the form of

$$\mathbb{P}_{\mathcal{A}}(\sigma|\mathbf{P}) = \frac{1}{C(\mathbf{P})} \prod_{\pi \in \mathbb{S}_K} \mathbb{P}_{\mathcal{A}}(\sigma|\pi, \mathbf{P}) \tag{1}$$

where $C(\mathbf{P}) = \sum_{\sigma \in \mathbb{S}_K} \prod_{\pi \in \mathbb{S}_K} \mathbb{P}_{\mathcal{A}}(\sigma|\pi, \mathbf{P})$ is the normalization factor. This model can be written in a more convenient form as follows. Let us denote the binary matrix by $\mathbf{D}^{\sigma,\pi} = \left[d_{i,j}^{\sigma,\pi}\right]_{1 \leq i,j \leq K}$ whose component $d_{i,j}^{\sigma,\pi} = 1$, if the sorting at hand compares o_i to o_j, when π was the initial ordering, and σ is its output. In addition, we need their sum over all orderings, that is, $\mathbf{D}^{\sigma} = \left[d_{i,j}^{\sigma}\right]_{1 \leq i,j \leq K}$, where $d_{i,j}^{\sigma} = \sum_{\pi \in \mathbb{S}_K} d_{i,j}^{\sigma,\pi}$, and the sum is meant elementwisely. With this notation, the model can be written as

$$\mathbb{P}_{\mathcal{A}}(\sigma|\mathbf{P}) = \frac{1}{C(\mathbf{P})} \prod_{\pi \in \mathbb{S}_K} \prod_{i=1}^{K} \prod_{j \neq i}^{K} p_{i,j}^{d_{i,j}^{\sigma,\pi}} = \frac{1}{C(\mathbf{P})} \prod_{i=1}^{K} \prod_{j \neq i}^{K} p_{i,j}^{d_{i,j}^{\sigma}},$$

[1] Department of Computer Science, Paderborn University, Germany, email: {adilp,busarobi}@upb.de

which is a special case of the log-linear model over the symmetric group because the probabilities can be written as a linear function of the logarithm of the parameters. Note that the key quantity in the model is $\mathbf{D}^\sigma$ which we shall compute in a closed form when insertion sort algorithm is used.

3 Insertion sort case

To make the definition of our model given in (1) complete, we shall make use of the insertion sort denoted by $\mathcal{I}$ and show that in this case one can write the model in a closed form. Next we are going to focus on $\mathbf{D}^\sigma = \sum_{\pi \in \mathbb{S}_K} \mathbf{D}^{\sigma,\pi}$ where the sum is meant elmenetwisely. The following observation allows us to compute $\mathbf{D}^\sigma$ in a concise way.

Lemma 1. *Assume that $\sigma_{id} = (o_1, \ldots, o_K)$. For any $1 \leq i, j \leq K$, if $i < j$, then*

$$d_{i,j}^{\sigma_{id}} = \begin{cases} d_{i,j}^{\sigma_{id}} = \frac{K!}{2} & , \text{ if } i < j \\ d_{i,j}^{\sigma_{id}} = \binom{K}{b_{i,j}+2}(K - b_{i,j} - 2)! b_{i,j}! & , \text{ if } j < i \\ 0 & , \text{ otherwise,} \end{cases}$$

where $b_{i,j} = i - j - 1$. Furthermore, for any $\sigma \in \mathbb{S}_K$, we have $\mathbf{D}^\sigma = \mathbf{D}^{\sigma_{id}}(\sigma)$ where $\mathbf{D}^{\sigma_{id}}(\sigma)$ is the permutation of the rows/columns of the matrix $\mathbf{D}^{\sigma_{id}}$ according to σ.

4 Parameter estimation

The maximum likelihood (ML) principle cannot be applied directly to our model, because the normalization factor $C(\mathbf{P})$ cannot be written in a closed form in terms of the model parameters. Therefore, we opted for using the Generalized Iterative Scaling (GIS) procedure [3]. Fist, assume that we are given a set of observations $\{\sigma_1, \cdots, \sigma_n\}$ for which the corresponding empirical frequencies are $\widehat{p}_i = \#\{i \in [K] : \sigma_i = \sigma_{\downarrow j}\}/n$, where $\sigma_{\downarrow j}$ is jth ordering according to some fixed ordering over the set of orderings, e.g. Lehmer code. Then the GIS procedure seeks to find a parameter estimate for which

$$\sum_{\ell=1}^{K!} p'_\ell d_{ij}^{\sigma_{\downarrow \ell}} = \sum_{\ell=1}^{K!} \widehat{p}_\ell d_{ij}^{\sigma_{\downarrow \ell}}$$

for all $i \neq j$ where $p'_\ell = \mathbb{P}_{\mathcal{A}}(\sigma_{\downarrow \ell}|\mathbf{P}')$ for some $\mathbf{P}'$. According to [3], $\mathbf{P}'$ is the ML estimate for $\mathbf{P}$, however we have reciprocal constraints which needs to be taken into account in our case. Note that the GIS method requires to compute a $K!$-long vector which can be alleviated based on a simple approximation technique.

5 Sampling the model

The model given in (1) can be sampled by using MCMC based on the fact that one can compute the acceptance ratio as

$$\log \frac{\mathbb{P}_{\mathcal{A}}(\sigma|\mathbf{P})}{\mathbb{P}_{\mathcal{A}}(\sigma'|\mathbf{P})} = \sum_{i=1}^{K} \sum_{j \neq i}^{K} (d_{i,j}^{\sigma} - d_{i,j}^{\sigma'}) \log p_{i,j} \ .$$

This allows us to make use of the Metropolis-Hastings (MH) algorithm. We use Mallows model [5] as proposal distribution. The pseudo-code of the sampling is given in Algorithm 1. The center ordering of the Mallows model $\mathbb{P}(.|\phi, \sigma)$, that is denoted by σ is always set to the current ordering σ_{i-1} (see line 5). In this case, it is easy to verify that the stationary distribution of the Markov chain is indeed $\mathbb{P}_{\mathcal{A}}(\sigma|\mathbf{P})$, because the Mallows model is symmetric, in a sense that $\mathbb{P}(\sigma|\phi, \sigma') = \mathbb{P}(\sigma'|\phi, \sigma)$, and it assigns positive probability to every ordering when $\phi > 0$. Therefore the detailed balance condition is satisfied, and moreover the ergodicity of the chain is also ensured.

Algorithm 1 Metropolis-Hastings algorithm with Mallows proposal

1: **procedure** MH(T, ϕ)
2: Select initial ordering σ_0
3: $\mathcal{D} = \emptyset$
4: **for** $i = 1 \to T$ **do**
5: $\sigma_i \sim \mathbb{P}(.|\phi, \sigma_{i-1})$ ▷ Proposal form Mallows model
6: Compute $q_i = \sum_{i=1}^{K} \sum_{j \neq i}^{K} (d_{i,j}^{\sigma_i} - d_{i,j}^{\sigma_{i-1}}) \log p_{i,j}$
7: Accept σ_i with probability $\min(1, \exp(q_i))$
8: $\mathcal{D} = \mathcal{D} \cup \{\sigma_i\}$
9: **end for**
10: **return** $\mathcal{D}$
11: **end procedure**

6 Related work

Biernacki and Jacques [1] studied a sorting-based ranking distribution which can be written in form of

$$\mathbb{P}_{\mathcal{A}}(\sigma|\mathbf{P}) = \sum_{\pi \in \mathbb{S}_K} \mathbb{P}_{\mathcal{A}}(\sigma|\pi, \pi', \mathbf{P}) \mathbb{P}(\pi), \tag{2}$$

where $\mathbb{P}(\pi)$ is the distribution over the initial orderings and $\mathbb{P}_{\mathcal{A}}(\sigma|\pi, \pi', \mathbf{P})$ is a noisy sorting model, that slightly differs from ours. First, they assume stationary noise for each pair, second, a pair of elements to be compared counts as discordant only if their ordering deviates from their ordering with respect to π'. The model fitting is a challenging task even if one assumes uniform distribution over the latent variables, i.e. $\mathbb{P}(\pi) = 1/K!$. Biernacki and Jacques applied an Expectation Maximization approach for maximizing the likelihood, and they conducted some numerical experiments with limited number of alternatives ($K < 15$). Note that this generative model is easy to sample, i.e. a sample from the model can be obtained by generating an initial order π, and then running the noisy sorting model with π.

Meek and Meila [6] investigated the noisy sorting model based on merge sort algorithm, that is, $\mathbb{P}_{\mathcal{M}}(.|\pi, \mathbf{P})$. They proposed a dynamic programming-based algorithm to estimate the pairwise matrix $\mathbf{P}$ for fixed π, and a heuristic search for the more general case when π is arbitrary.

REFERENCES

[1] Christophe Biernacki and Julien Jacques, 'A generative model for rank data based on insertion sort algorithm', *Comput. Stat. Data Anal.*, **58**, 162–176, (February 2013).

[2] V. Csiszár, *Statistical analysis of random permutations*, Ph.D. dissertation, ELTE, 2008.

[3] J. N. Darroch and D. Ratcliff, 'Generalized iterative scaling for log-linear models', *Ann. Math. Statist.*, **43**(5), 1470–1480, (10 1972).

[4] Persi Diaconis and Bernd Sturmfels, 'Algebraic algorithms for sampling from conditional distributions', *Ann. Statist.*, **26**(1), 363–397, (02 1998).

[5] C. Mallows, 'Non-null ranking models', *Biometrika*, **44**(1), 114–130, (1957).

[6] Christopher Meek and Marina Meila, 'Recursive inversion models for permutations', in *Advances in Neural Information Processing Systems 27*, eds., Z. Ghahramani, M. Welling, C. Cortes, N.d. Lawrence, and K.q. Weinberger, 631–639, Curran Associates, Inc., (2014).

[7] Balázs Szörényi, Róbert Busa-Fekete, Adil Paul, and Eyke Hüllermeier, 'Online rank elicitation for plackett-luce: A dueling bandits approach', in *NIPS*, pp. 604–612, (2015).