

Adapting to a realistic decision maker: experiments towards a reactive multi-objective optimizer

Paolo Campigotto and Andrea Passerini

DISI - Dipartimento di Ingegneria e Scienza dell'Informazione,
Università di Trento, Italy
{campigotto, passerini}@disi.unitn.it

Abstract. The interactive decision making (IDM) methods exploit the preference information from the decision maker during the optimization task to guide the search towards favourite solutions. This work measures the impact of inaccurate and contradictory preference information on the quality of the solutions generated by the IDM methods. The investigation is done in the context of the BC-EMO algorithm, a recently proposed multi-objective genetic algorithm.

1 Introduction

Modeling real world problems often generates optimization tasks involving multiple and conflicting objectives. Because the objectives are in conflict, a solution simultaneously optimizing all of them does not exist. The typical approach to the multi-objective optimization problems (MOOPs) consists of searching a set of trade-off solutions, called the *Pareto-optimal* set, for which any single objective cannot be improved without compromising at least one of the other objectives. Usually, the size of the Pareto-optimal set is large or infinite and the decision maker (DM) cannot tackle the overflow of information generated when analyzing it entirely. In this scenario, the *interactive* decision making (IDM) technique comes to the rescue. It assumes that the optimization expert (or the optimization software) cooperates with the DM. Through the interaction, the search process can be directed towards the DM favorite Pareto-optimal solutions and only a fraction of the Pareto-optimal set needs to be generated. Several IDM approaches have been developed to aid the DM in identifying his preferred solution [4]. A recent method is the Brain-Computer Evolutionary Multi-Objective Optimization (BC-EMO) algorithm [1], a genetic algorithm that learns the preference information of the decision maker (formalized as a value function) by the feedback received when the DM evaluates tentative solutions. Based on this feedback, the value function is refined, and it is used to modify the population of the genetic algorithm.

In the experiments presented in [1], the preference information of the DM is assumed accurate: the algorithm is tested using an ideal user providing consistent and correct answers to each question generated during the interactive process.

However, in many concrete applications assuming non-contradictory and consistent feedback from the decision maker is rather unrealistic. An important issue is therefore studying the quality of the solutions generated by an IDM algorithm as a function of the accuracy of the preference information provided by the DM. This contribution explores the robustness of the BC-EMO algorithm in two noisy scenarios emulating possible inaccuracies in the DM feedback.

2 The BC-EMO algorithm

The goal of the BC-EMO algorithm consists of learning the non-dominated solution preferred by the decision maker. To fulfill this scope, BC-EMO learns a value function from the preference information provided by the DM by using the support vector ranking, a supervised machine learning technique that learns to rank the input data. Training examples consist of pairwise comparisons of non-dominated solutions which are turned into ranking constraints for the learning algorithm. No specific assumptions are made about the form of the DM value function: BC-EMO has a tuning phase selecting the most appropriate kernel (i.e., similarity measure) in order to best approximate the targets, allowing it to learn an *arbitrary* value function provided enough data are available. Furthermore, support vector ranking allows to effectively deal with noisy training observations thanks to a regularization parameter C trading-off data fitting with complexity of the learned model. This aspect motivates the analysis of the noise robustness of BC-EMO performed in this contribution.

The learned value function is used to order the current population during the *selection* phase of the BC-EMO algorithm, where a sub-population is selected for reproduction on the basis of fitness (i.e., quality of the solutions). In particular, the BC-EMO selection procedure consists of:

1. collecting the subset of non-dominated individuals in the population;
2. sorting them according to the learned value function;
3. appending to the sorted set the result of repeating the procedure on the remaining dominated individuals.

The procedure is guaranteed to retain Pareto-optimality regardless of the form of the learned value function. Any evolutionary multi-objective algorithm (EMOA) that needs comparisons between candidate individuals can be equipped with the BC-EMO selection procedure (replacing or integrating the original selection procedure). Following [1] we focused on the NSGA-II [2] EMOA. The overall BC-EMO approach consists of 3 steps:

1. *initial search phase*: the *plain* EMOA selected is run for a given number of generations collecting the final population P_1 ;
2. *training phase*: using P_1 as initial population, a specific number of training iterations are executed to learn the value function V by interacting with the DM. The final population obtained (P_2) is collected;

3. *final search phase*: the selected EMOA equipped with the BC-EMO selection procedure is run for a given number of generations, using P_2 as initial population and producing the final ordered population.

Each training iteration alternates a refinement phase, where the DM is queried for feedback on candidate solutions and the value function is updated according to such feedback, with a search phase, where the EMOA equipped with the BC-EMO selection procedure is run for a given number of iterations. The training phase is executed until the maximum number of training iterations or the desired accuracy level are reached.

3 Experimental results

Incorrect preference information can be due to occasional inattention of the DM, or by his embarrassment when required to compare too similar solutions. To represent these situations, two models of inaccurate preference information are considered:

1. the probability P_i of incorrect feedback from the DM is a constant value $\gamma \in (0, 0.3]$;
2. the probability P_i of incorrect feedback from the DM increases with the similarity of the solutions to be compared.

In the second model, $P_i = \gamma \cdot \langle \mathbf{z}, \mathbf{z}' \rangle / \sqrt{\langle \mathbf{z}, \mathbf{z} \rangle \cdot \langle \mathbf{z}', \mathbf{z}' \rangle}$ where \mathbf{z} and \mathbf{z}' are the objective vectors of the solutions to be compared and γ is a constant value in the range $(0, 0.3]$.

We selected as a benchmark the DTLZ6 problem taken from [3], with the following polynomial value function: $0.05 \cdot z_2 z_1 + 0.6 \cdot z_1^2 + 0.38 \cdot z_2 + 0.23 \cdot z_1$, where z_1 and z_2 are the objectives to optimize. As shown in [1], this polynomial value function maps the Pareto-optimal front to a non-linear and disconnected surface for which a linear approximation fails to recover the desired solution. We thus evaluate the ability of the BC-EMO algorithm to both correctly select an appropriate non-linear kernel and learn the parameters of the resulting function in a noisy scenario.

Fig. 1 contains the results obtained for the two models of noise we considered. The performance of the algorithm is measured in terms of the approximation error w.r.t. the *gold standard* solution (y -axis) in function of γ (x -axis). The gold standard solution is obtained by guiding the algorithm with the true value function. The results are the median of 100 runs of the BC-EMO algorithm with a single training iteration. Different curves represent different numbers of training examples. The regularization parameter C of the support vector ranking was fixed to 1 in all experiments. Qualitatively similar results were obtained for higher values of C , including the value of $C = 100$ employed in [1] in an unnoisy scenario. Our aim here is to show a trend confirming the robustness of the algorithm rather than boost its performance to the limit. It is straightforward to include a fine tuning of the regularization parameter within the model selection phase in order to adapt it to the problem at hand.

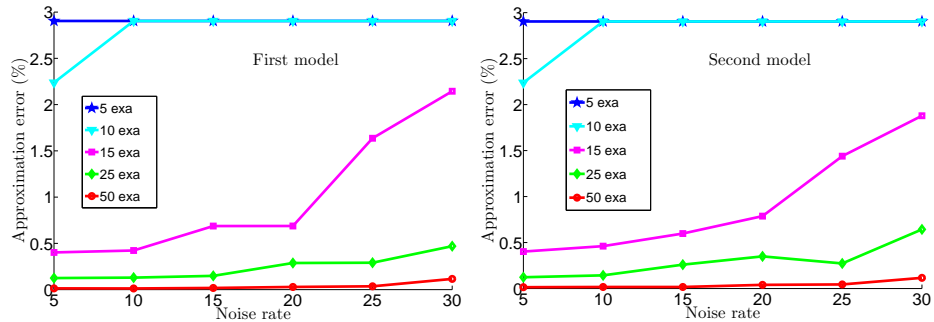


Fig. 1: Learning curves for an increasing number of training examples observed for the two models of inaccurate preference information. The y -axis reports the solution quality, while the x -axis contains the values of γ .

The algorithm shows a similar behaviour for both noise models. As expected, the performance tends to degrade when increasing the amount of noise, while it improves for an increasing number of training examples. However, the results tend to remain within 3% of the gold standard solution, which is the solution found by a linear approximation of the value function in an unnoisy scenario (see [1]). Indeed more than half of the times a linear kernel is incorrectly chosen when 5 noisy training instances are provided. Results rapidly improve with a larger number of examples, being basically insensitive to up to 30% of noise for 25 examples or more.

These preliminary experimental results are promising: the performance of the BC-EMO is robust in the presence of inaccurate preference information from the DM in a non-linear setting. Future work includes testing more complex models for the user value function and his uncertainty, modifying the learning phase in order to account for realistic patterns of noise, and developing an active learning strategy to minimize the required feedback.

References

1. R. Battiti and A. Passerini. Brain-computer evolutionary multi-objective optimization (BC-EMO): a genetic algorithm adapting to the decision maker. Technical Report DISI-09-060, DISI - Dipartimento di Ingegneria e Scienza dell'Informazione, Università di Trento, Italy. (Submitted for Journal publication)., 2009.
2. K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan. A fast elitist multi-objective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, 6:182–197, 2000.
3. K. Deb, L. Thiele, M. Laumanns, and E. Zitzler. Scalable multi-objective optimization test problems. In *Congress on Evolutionary Computation (CEC2002)*, pages 825–830, 2002.
4. K. Miettinen, F. Ruiz, and A.P. Wierzbicki. Introduction to Multiobjective Optimization: Interactive Approaches. In *Multiobjective Optimization: Interactive and Evolutionary Approaches*, pages 27–57. Springer-Verlag Berlin, Heidelberg, 2008.