




Learning Decision Variables in Many-Objective Optimization Problems

Artur Leandro da Costa Oliveira , René Gusmão  and André Britto 

Abstract—Traditional Multi-Objective Evolutionary Algorithms (MOEAs) have shown poor scalability in solving Many-Objective Optimization Problems (MaOPs). The use of machine learning techniques to enhance optimization algorithms applied to MaOPs has been drawing attention due to their ability to add domain knowledge during the search process. One method of this kind is inverse modeling, which uses machine learning models to enhance MOEAs differently, mapping the objective function values to the decision variables. The Decision Variable Learning (DVL) algorithm uses the inverse model in its concept and has shown good performance due to the ability to directly predict solutions closed to the Pareto-optimal front. The main goal of this work is to experimentally show the DVL as an optimization algorithm for MaOPs. Our results demonstrate that the DVL algorithm outperformed the NSGA-III, a well-known MOEA from the literature, in almost all scenarios with restriction on the number of objective functions with a high number of objectives.

Index Terms—Many-Objective Optimization, Machine Learning, Inverse Surrogate Models, Decision Variable Learning.

I. INTRODUCTION

A Multi-Objective Problem (MOP) is an optimization problem having two or more objective functions to be optimized. MOPs that have more than three objectives are called Many-objective Optimization problems (MaOPs), and the field that studies new solutions for these problems is called Many-Objective Optimization. It is known that MOEAs scale poorly in many-objective optimization problems [1]. To solve these complex problems, the combination of Optimization algorithms with machine learning techniques has been the subject of research in the last few years [2]. Machine learning methods can be used to extract knowledge and integrate into the optimization process. This knowledge can take different forms and result in different ways of integrating the knowledge into the metaheuristic.

There has been an effort in the area to develop techniques that provide approximate models to represent the original problem in a way that facilitates solving it, called meta-models. There are several meta-modeling methods, but they can be divided into two main categories: classic function transformation methods and surrogate modeling methods. The first category corresponds to the techniques that seek to represent the set of functions of the problem in only one, and the second seeks a new expression for each objective function

that is constructed based on the data obtained previously from the real objective function, called surrogates.

Despite the existence of related work on surrogates applied to multi- and many-objective optimization, they only explore learning from decision variables to estimate the objective functions. However, since in MaOPs we have objective vectors with higher dimensions, could it be possible to learn the behavior from objective functions to estimate the decision variables?; since it is possible to learn the behavior from decision variables using surrogates, is it possible to learn inverse functions and from that functions, to predict sub-optimal solutions to the problem?

The work done in [3] answers part of this question by using inverse surrogate models to generate more non-dominated solutions for the decision-maker (DM), based on the approximated Pareto set obtained by MOEAs at the end of evolutionary optimization, increasing the density of the Pareto set found. Some related work has been exploring the use of surrogates in recent years. In [4] a many-objective hybrid optimizer, called MOHO, that uses five constitutive algorithms and actively switches among them throughout the optimization process. Also, SAMaOEA [5], a reference vector guided evolutionary algorithm assisted by radial basis function (RBF) models. Furthermore, a surrogate-based evolutionary algorithm was proposed in [6] for expensive multi-objective optimization problems.

In [7] is proposed the Decision Variable Learning (DVL) algorithm that explores an inverse model approach in many-objective optimization. The DVL algorithm aims to design a model, using machine learning, to represent the relationship between objective functions and variables, using the objective vectors as input to estimate a set of solutions in decision space. In this process, reference points - which are ideal points, not guaranteed if they are possible - will be passed as input to the model. Thus, the model will return a set of solutions, with objective vectors as close as possible to those reference points.

These initial DVL results help us to answer part of our research question, however, there are still open topics that should be further explored to understand if it is possible to estimate the decision variables. The first topic is if a machine learning model can be trained to understand the relationship between objectives and decision variables of an optimization problem, and what is the error rate of this learning. The second topic is related to the understanding of the impact of training a model in different scenarios during an optimization process in comparison to the NSGA-III.

In this paper, we present an analysis through empirical experiments to validate the use of inverse surrogate models

Artur Leandro da Costa Oliveira is with Graduate Program on Computer Science, Federal University of Sergipe e-mail:arturleandrocosta@gmail.com.

René Gusmão and André Britto are with Department of Computing, Federal University of Sergipe e-mail:rene@dcomp.ufs.br, andre@dcomp.ufs.br.

in Many-Objective Optimization. Here, DVL algorithm is the inverse model used. The experiments are intended to demonstrate if it is possible to learn decision variables from objective functions, and validate the DVL as an algorithm for MaOPs, analyzing the impact of different machine learning models have on the algorithm. The experiments will be performed using MaOPs benchmark problems, and the results obtained will be compared with the results of the NSGA-III algorithm.

The main contributions of our work are: (i) validation of machine learning algorithms to be used for estimating the Pareto optimal set of optimization problems; (ii) comparison of different classical machine learning algorithms as inverse models; and, (iii) analysis of the performance of the DVL algorithm in solving different scenarios of optimization problems.

The remaining sections of this paper are organized as follows: Section II describes the related works, Section III presents the algorithm studied in this work, Section IV describes how the experiments were executed with the results found, and finally, Section V concludes the paper.

II. RELATED WORK

According to our research, the first attempt of constructing an inverse modeling method is made in [3], called Pareto Estimation (PE). Where an inverse functional mapping from the Pareto Front (PF) to the Pareto Set (PS) was built based on the non-dominated solutions set obtained by the execution of an MOEA at the end of evolutionary optimization. This model is used to generate additional non-dominated solutions, thereby enhancing the density of the solutions set. The motivation is that in many MOPs, principally in MaOPs, the decision-maker is not satisfied with the PS obtained by the MOEA. In this work, it should be noted that the inverse model is not used during the optimization process differently from the DVL algorithm.

In [8] the PE method is extended to improve the density of available non-dominated solutions in multi-objective multimodal problems. This kind of problem has multiples decision vectors that map to identical objective vectors on the Pareto front. In this case, the authors propose to subdivide the objective space, by using a clustering algorithm, into different C_m clusterings. So an ANN is trained for each cluster of solutions to identify the map $\tilde{F}_{\tilde{P}C_m} : \tilde{P} \rightarrow C_m$. In this work, it is noted that the quality of the resulting inverse model depends heavily on the training samples used. In [9], it is proposed the MAEA-GD/RD, an optimization algorithm based on the MAEA-gD with sample redistribution, which reallocates the samples in the current population in objective space for training the inverse model. Doing this, more samples are present in regions where higher quality topological information is more likely to be obtained and used to improve the identification of \tilde{F}_P .

In [10] is proposed the IM-MOEA algorithm, motivated by the PE method and the EDAs, provides a different way of using inverse models. In the IM-MOEA, the inverse models are inserted internally in the optimization algorithm and then used to create offspring solutions by sampling the objective space. They propose a decomposition of the inverse function, also reducing the number of inverse models, and a particular

reproduction operator who benefits from the inverse model. The purpose of their method is to use the inverse models to increase the density of solutions in the preferred regions at a low computational cost.

The main contributions of the literature on inverse models are based on using the knowledge extracted from the inverse model at the end of the optimization process, or internally creating offspring solutions. These strategies limit the use of inverse models in generating good solutions at the beginning of the optimization process, decreasing the evolutionary time to reach an optimal set. In a different manner, the DVL algorithm uses the inverse model to generate, at each iteration, an entirely new solution set. Using the knowledge from the previous iteration, the DVL algorithm produces solutions as close as possible to the Pareto optimal set.

III. DECISION VARIABLE LEARNING (DVL)

The Decision Variable Learning algorithm [7] aims to use machine learning models to generate an approximated Pareto optimal set using information from samples of the objective space and a set of reference points. Differently from the Pareto Estimation (PE), the DVL is not a method used after the MOEA, instead, it is an algorithm that uses the inverse model approach into the algorithm to solve multi-objective problems. Although in the elaboration of the DVL the authors were not aware of the studies relative to the inverse model, the DVL has the theoretical basis of the PE method. But, instead of using the inverse model to increase the number of non-dominated solutions, the DVL directly infers an approximation of the Pareto optimal set.

DVL is based on three main steps, the first one is the creation of an initial sample dataset, that contains solutions with their respective objective functions. This initial sample generation is provided by the LHS method, which generates well-distributed samples over the search space. The second step is the training of the inverse method, a machine learning model is used for this purpose, which is trained with the dataset previously generated. The idea that supports this modeling is that, for continuous MOPs, it can be deduced from the Karush-Kuhn-Tucker optimality conditions that the Pareto optimal set is a piecewise continuous in the decision variables [3]. Depending on the machine learning model, M_n models are created, and each one estimates only one variable, so each solution generated in this process will be composed of variables estimated by different models. Another strategy is to use only one model M that can be used to estimate the entire decision variable set. The last step is the prediction of new solutions next to the Pareto optimal set. The objectives set used as input for this step is the same as the PE method, which is based on the method in [11], that places points on a normalized hyper-plane equally inclined to all objective axes and has an intercept of one on each axis. The algorithm was projected to work iteratively, where, at each iteration, the training of the machine learning model and prediction of a new population is executed. These steps are performed using a set of solutions that is incremented at each iteration. The purpose of this iterative approach is to improve the quality of

the dataset and thereby improve future estimations since the dataset will have solutions closer to the reference points.

One real-world application for DVL is the Traffic Lights Signaling [12]. Traffic light signaling optimization is one way of improving traffic efficiency. It relies on optimizing the traffic-light cycle, which is the sequence of states of a traffic light. A traffic simulator can be used to obtain data on traffic flow through traffic lights synchronization, which provides all the data necessary to simulate a traffic flow. The problem can be modeled as a MaOPs, since the decision variable time of each traffic-light cycle, and the objective functions are metrics obtained through the simulation such as quality measures such as travel and stopped time, the number of vehicles that arrive at their destination, the overall average speed, among others. In this problem even for a single intersection there can be no obvious optimal solution. Furthermore, the simulation is costly. This real optimization problem has an expensive objective function and constraints, providing a new scenario for studying the DVL inverse modeling.

IV. EXPERIMENTS AND RESULTS

In this section, we evaluate the DVL algorithm through empirical experiments that have two main motivations: first, it is necessary to validate the possibility to learn decision variables from objective functions, for this, we propose the Regression experiment. Secondly, in the optimization experiment, we validate the impact that different learning models have on the algorithm. Also, we validate the DVL as an algorithm for Many-objective Optimization in comparison with a state-of-the-art MOEA, the NSGA-III.

In our experiments, we used the first seven test problems from the DTLZ benchmark test suite [13]: DTLZ1 to DTLZ7, which are multiobjective problems for optimization that have different configurations and shapes of the Pareto-optimal front. The problems are parameterized from a variable k , where $k = n - m + 1$, being n the number of variables and m the number of objectives of the problem. Two different instances were defined for each problem, one with 3 objectives and 12 variables and the other with 10 objectives and 12 variables. Although a problem with 3 objective is not considered a MaOPS, we adopted this scenario to analyze the scalability performance of the compared algorithms. By the DTLZ definition, the complexity of 10 objectives is lower than the complexity of 3 objectives. However, the variation of the objective number allows the identification of the influence of the objectives in the learning model. In a previous work [7], the variation in decision variables showed no impact on algorithm performance, so the number of variables is set to 12 in both cases. All the algorithms in the experiments were subjected to the same conditions, and our experiments aim to make a comparative analysis between them.

Two different test configurations are described and analyzed. The first one is called the regression experiment, and the second is the optimization experiment. The parameter settings and the performance indicators used in each test are described at the beginning of each test subsection. In all experiments, the algorithms are executed independently 20 times for each configuration.

In both experiments, Friedman's test was used to check whether there are statistically significant differences considering all algorithms. Then, the Wilcoxon rank-sum test is adopted to compare the differences between pairs of configurations. It was adopted 0.05 of the significance level. As pointed out in [14] assumptions such as independence, normality, and homoscedasticity are most probably violated when analyzing the performance of stochastic algorithms based on computational intelligence. Thus, normally non-parametric tests are used. The results of the statistical test were used to compute the number of wins, ties and losses of an algorithm. An algorithm wins when it obtains the higher mean values of the metric and its results are statistically different to any other algorithms' result. In these cases, a loss is computed for the other algorithms. It is a tie when there is no statistical difference between the algorithm with the higher mean and some other algorithm.

In the default implementation of the DVL algorithm, n models are created to estimate each decision variable. Different from the strategy used in [3] where a single model is used to estimate the entire decision variable set. Because of the dependence between the input and the outputs that can occur in optimization problems, the two strategies are applied in our experiments.

A. Learning Models

A set of different regression machine learning models, well known in the literature, are used in our experiments to evaluate the impacts of these models in learning the decision variables. The Linear model, an artificial neural network, more precisely a Multilayer Perceptron (MLP), a Random Forest regression (RFR), and a Support Vector Regression (SVR). Each one has unique characteristics that make them perform well in certain types of problems. The implementation and the parameters of each model were defined by Scikit-learn, a python module that integrates a wide range of state-of-the-art machine learning algorithms. Different from the default implementation of each model in the Scikit-learn, were used as parameters: (11, 11, 11) for the number of neurons in the i th hidden layer, not counting the input and output layers of the MLP. 100 for the number of trees in the forest of the RFR, and 0.1 for the regularization parameter of the SVR. These values of the parameters described were the best found by empirical analysis of running an experiment with the chosen models in a smaller test case with different parameters empirically and randomly chosen. In this smaller test case, it was adopted the same methodology described in this section.

With few exceptions, Machine Learning algorithms do not perform well when the input contains numerical attributes with significantly different scales. It is the case of the objective values of the DTLZ benchmark test suit. Therefore, for each model was used the standardization transformation which is an algorithm of feature scaling. The standardization subtracts the mean value (so standardized values always have a zero mean), and then it divides by the variance so that the resulting distribution has unit variance. This transformation causes a better performance in our experiments.

To validate and analyze the strategy of using one model to predict the entire decision variable set or using one model to each decision variable, in our experiments, we applied the multioutput regression in the models to obtain knowledge from the dependence between inputs and outputs. The exception is the SVR model, which implementation in Scikit-learn does not have support to multioutput regression. The models with this strategy are denoted with an "MO" before the model name in our results.

B. Regression Experiment

The first experiment is motivated to answer if it is possible to learn decision variables from objective functions. The task of understanding and predicting the variables of the DTLZ benchmark test problems can be classified as supervised learning since we have the input and output for the learning algorithm and can be classified as a regression task since we want to predict a numerical value. This experiment is based on two phases: training and testing. First, the model is trained by the execution of the DVL algorithm. The generated samples and modeling strategy follow the default flow of the algorithm. Secondly, in the testing step, a subset of the Pareto-optimal front of each DTLZ problem serves as input to the model for prediction. The DTLZ benchmark test suit allows the generation of the Pareto-optimal front of each problem analytically. The key idea of this experiment is to analyze if the inverse modeling of the DVL algorithm can correctly predict the optimal set of a problem. In a real optimization problem, the optimal front is unknown, but the purpose of this experiment is to validate the inverse modeling strategy trained with solutions far from the optimal front.

Each model in this experiment is trained with an initial number of solutions generated through the LHS algorithm. The size of the initial sample solutions is defined as 10000. For each DTLZ problem, approximately 5000 solutions of the Pareto-optimal front were generated for the experiment. The values of these Pareto-optimal front were obtained through the execution of diverse MOEAs without restriction and a post-processing routine to diversify these solutions. The objective function values of these generated solutions are computed and passed as input for each model to predict the decision variables. The predicted values of each model are evaluated to the objective space. The mean-squared error (MSE) metric is used to make empirical comparisons between the results of each model. The MSE is the average squared difference between the predicted optimal front and the Pareto-optimal front values.

The results of the regression experiment for 3 and 10 objectives are shown in Table I. The wins or ties are highlighted in bold. The model with the lower MSE value was more successful in predicting decision variables. The results obtained in this experiment show that different models have better performance on different types of problems. Also, the number of objectives has interference in the results of MSE on each model for each problem. Although the MSE obtained has relatively low values, it is necessary to evaluate the predicted decision variables and the objectives produced

by these variables in terms of divergence and convergence produced. For that, we made a visual comparison with the results obtained in the experiment and the Pareto-optimal front values of each problem.

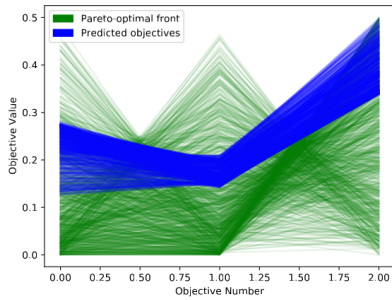
Fig. 1 shows the visual comparison of the regression experiment. For each problem, the model that achieves the best value of MSE is used in comparison with the Pareto-optimal front. Different models were used for comparison which indicates that the use of a specific model is linked to the problem that will be solved. Each pair of charts represent the comparison for 3 and 10 objectives respectively. In each chart, the green color lines represent the Pareto-optimal front, the blue lines the objectives from the predicted decision variables. Every line in the chart details an objective function vector, the x-axis is the position in the vector, and the y-axis is the objective function value contained in the vector position.

In all charts, the models used achieved a high convergence, i.e. the values found are very close to the ideal. Only in the DTLZ6 problem, Fig. 1k and Fig. 1l, that the predicted objectives are a little far away from the ideal. DTLZ6 has a degenerate Pareto-optimal front as like DTLZ5, besides that, the DTLZ6 has many locals Pareto fronts which makes it difficult to converge to the Pareto-optimal front. In Figs. 1a, and 1e, respectively, DTLZ1, and DTLZ3 problems, the models could not find all parts of the Pareto-optimal for 3 objectives. It occurred mainly due to the fact of the high search space presented in these problems. Also, they have many local Pareto-optimal fronts, which makes it difficult to obtain a good divergence on the predicted values. In the rest of the problems the predicted values obtained have performed well in divergence and convergence of values. In these problems, the shape of the region formed by the predicted values is quite similar to the shape of the Pareto-optimal front. It is important to note that the models had a greater divergence of values in the problems with 10 objectives, demonstrating a shape closer to the ideal. This is mainly due to the increased number of features used for training which makes the models perform better than with 3 objectives. Concerning the strategy of using 1 or n models to estimate each decision variable, the results show that both strategies can be used. In different optimizations problems, each one gets the best results.

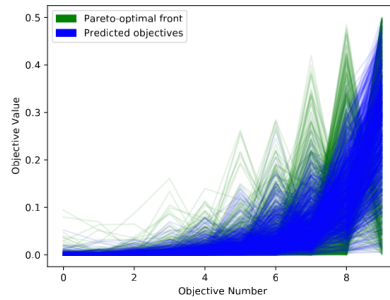
The results found by the regression experiment are quite promising. We show that different models could learn the decision variables by a set of objective function values. Even in more restricted scenarios, the models achieved good results. For the scope of this work, the models and techniques used were enough to validate the learning of decision variables.

C. Optimization Experiment

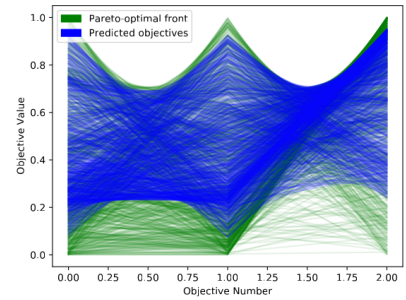
In this experiment, each algorithm runs until it reaches a stop condition, in the experiments we use the number of evaluations of the objective function as a stop condition. A set of a maximum number of evaluations to our experiment is defined as (250, 500, 1000, 1500, and 10000), all algorithms run with the same number of calls to the objective function. This validation set is used to simulate different scenarios of restrictions in the use of the objective function. In this set, we



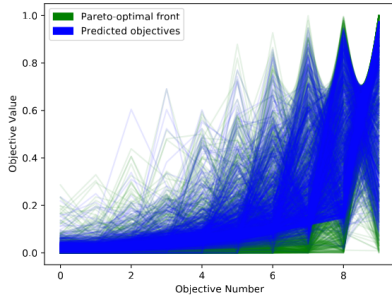
(a) DTLZ1 - 3 objectives



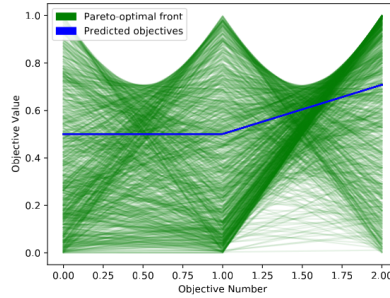
(b) DTLZ1 - 10 objectives



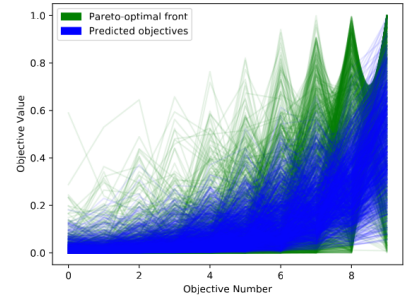
(c) DTLZ2 - 3 objectives



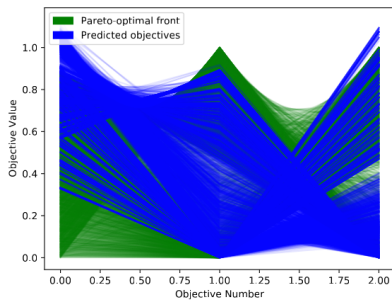
(d) DTLZ2 - 10 objectives



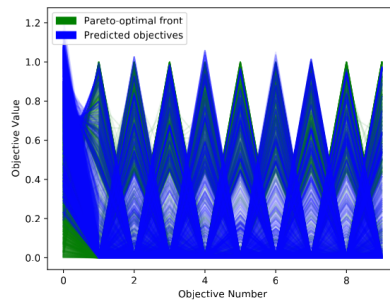
(e) DTLZ3 - 3 objectives



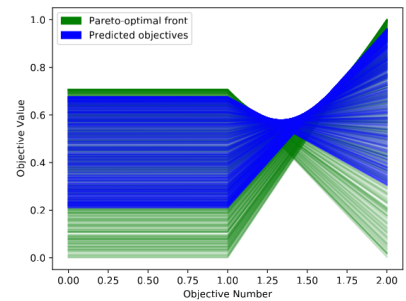
(f) DTLZ3 - 10 objectives



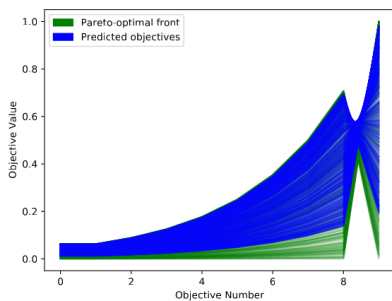
(g) DTLZ4 - 3 objectives



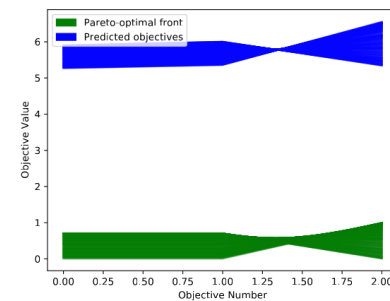
(h) DTLZ4 - 10 objectives



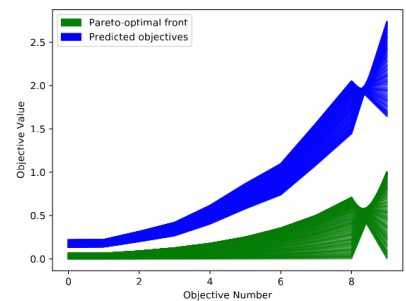
(i) DTLZ5 - 3 objectives



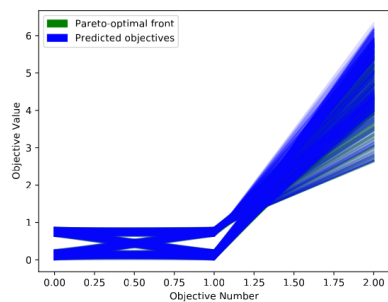
(j) DTLZ5 - 10 objectives



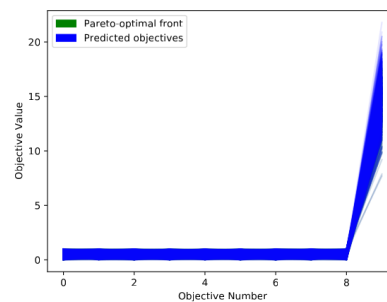
(k) DTLZ6 - 3 objectives



(l) DTLZ6 - 10 objectives



(m) DTLZ7 - 3 objectives



(n) DTLZ7 - 10 objectives

Fig. 1. Visual Comparisons for the DTLZ benchmark. (a) SVR model, (b) RFR model, (c) SVR model, (d) RFR model, (e) SVR model, (f) RFR model, (g) RFR model, (h) RFR model, (i) SVR model, (j) RFR model, (k) MO-MLP model, (l) MO-Linear model, (m) MO-Linear model, and (n) MO-Linear model.

TABLE I
 STATISTICAL RESULTS (MEAN AND STANDARD DEVIATION) OF THE MSE VALUES FOR 3 AND 10 OBJECTIVES.

Obj.	Prob.	MO-Linear	SVR	MO-MLP	MO-RFR	Linear	MLP	RFR
3	DTLZ1	(1.42±0.01)e-2	(1.41±0.00)e-2	(1.84±0.26)e-2	(3.60±0.87)e-2	(1.42±0.01)e-2	(2.44±0.63)e-2	(3.62±0.21)e-2
3	DTLZ2	(4.49±0.02)e-3	(2.20±0.14)e-3	(3.97±0.99)e-3	(7.91±1.96)e-3	(4.50±0.02)e-3	(4.37±1.35)e-3	(3.69±0.37)e-3
3	DTLZ3	(1.43±0.00)e-2	(1.41±0.00)e-2	(1.98±0.39)e-2	(4.59±0.95)e-2	(1.43±0.01)e-2	(2.75±0.66)e-2	(3.80±0.30)e-2
3	DTLZ4	(1.20±0.04)e-2	(1.31±0.16)e-2	(3.67±1.27)e-2	(9.17±1.01)e-3	(1.19±0.05)e-2	(5.89±2.32)e-2	(9.39±1.26)e-3
3	DTLZ5	(8.58±0.02)e-3	(8.21±0.11)e-3	(9.98±1.67)e-3	(1.42±0.14)e-2	(8.59±0.02)e-3	(1.14±0.11)e-2	(1.03±0.03)e-2
3	DTLZ6	(1.11±0.03)e-1	(2.68±0.03)e-1	(9.41±3.83)e-2	(1.10±0.07)e-1	(1.11±0.04)e-1	(1.17±0.67)e-1	(1.20±0.08)e-1
3	DTLZ7	(2.91±0.75)e-4	(2.83±0.08)e-1	(5.08±1.97)e-2	(7.65±0.26)e-2	(3.00±0.75)e-4	(6.17±1.89)e-2	(6.77±0.72)e-2
10	DTLZ1	(6.39±0.00)e-2	(6.54±0.13)e-2	(6.45±0.07)e-2	(5.95±0.10)e-2	(6.39±0.00)e-2	(5.80±0.14)e-2	(1.76±0.33)e-2
10	DTLZ2	(3.53±0.01)e-2	(3.47±0.01)e-2	(3.07±0.21)e-2	(1.80±0.03)e-2	(3.52±0.01)e-2	(7.07±0.39)e-3	(1.32±0.04)e-3
10	DTLZ3	(6.39±0.00)e-2	(6.41±0.04)e-2	(6.44±0.04)e-2	(6.47±0.15)e-2	(6.39±0.00)e-2	(5.87±0.13)e-2	(1.90±0.19)e-2
10	DTLZ4	(5.41±0.01)e-2	(5.37±0.03)e-2	(6.87±0.16)e-2	(4.33±0.06)e-2	(5.41±0.02)e-2	(5.93±0.12)e-2	(3.91±0.04)e-2
10	DTLZ5	(5.69±0.00)e-2	(5.69±0.00)e-2	(5.75±0.03)e-2	(6.14±0.05)e-2	(5.69±0.00)e-2	(5.82±0.05)e-2	(6.17±0.04)e-2
10	DTLZ6	(6.81±0.07)e-2	(3.01±0.08)e-1	(1.45±0.29)e-1	(9.41±0.48)e-2	(6.83±0.08)e-2	(1.01±0.22)e-1	(8.25±0.18)e-2
10	DTLZ7	(4.99±0.59)e-5	(9.73±0.80)e-2	(3.10±1.35)e-2	(4.30±0.05)e-2	(4.74±0.64)e-5	(4.97±1.54)e-3	(1.98±0.30)e-3

classify as a higher restriction level a number of validations lower or equal to 1000. The empirical comparisons between the results are made by the use of Hypervolume (HV), a performance indicator that assesses both convergence and divergence of the solutions. Let y^* be a reference point dominated by all the Pareto front solutions, and P the population resulting of a MOEA. The HV of P is the calculus of the volume of the region that dominates y^* and is dominated by P . The HV values presented in this work are all normalized to $[0, 1]$ by dividing $\prod_{i=1}^m y_i^*$. In this work, the following reference points y^* are used for each problem: for DTLZ1, $y^* = (300.0, 300.0, \dots, 300.0)$; DTLZ2 and DTLZ4, $y^* = (2.0, 2.0, \dots, 2.0)$; DTLZ3, $y^* = (700.0, 700.0, \dots, 700.0)$; DTLZ5, $y^* = (4.0, 6.0, \dots, 2(m+1))$; DTLZ6 and DTLZ7, $y^* = (10.0, 18.0, \dots, 2(4m+1))$ for m number of objectives.

The DVL algorithm needs four parameters to be redefined for execution. The value c of the number of closest solutions of the reference points used for training. A k value for the initial number of population used by the LHS. A value i of the maximum number of iterations and a value ϵ of the difference between the HV of the population generated. In this experiment the c value is fixed in 200 and the ϵ is set to 0.001, the i and the k have different values for each configuration of number objectives vs. number of evaluations. For 3 objectives: $i = 1$ and $k = 159$ for 250 number of evaluations; $i = 3$ and $k = 227$ for 500; $i = 7$ and $k = 363$ for 1000; $i = 9$ and $k = 681$ for 1500 and $i = 14$ and $k = 8726$ for 10000. For 10 objectives: $i = 1$ and $k = 50$ for 250 number of evaluations; $i = 2$ and $k = 112$ for 500; $i = 4$ and $k = 132$ for 1000; $i = 5$ and $k = 464$ for 1500 and $i = 12$ and $k = 7388$ for 10000.

The parameters used in NSGA-III were defined by Pymoo, a Python framework for optimization. For the simulated binary crossover, the distribution index is set to $\eta c = 30$, and the crossover probability $pc = 1.0$. For the polynomial mutation, the distribution index and the mutation probability are set to $\eta m = 20$ and $pm = 1/n$, respectively, as recommended in [15]. The population size used was 91 and 220 for the problem with 3 and 10 objectives, respectively, and the individual has a length of 12 for both objective sizes.

Table II summarizes the results of the statistical comparison in terms of wins, draws, and losses. It is possible to note the DVL superiority in problems with 10 objectives by winning

34 of 35 scenarios. With the MLP model having 24 wins in comparison to the others. In the experimentation with 3 objectives, the NSGA-III wins a total of 14 in 35, and the combination of all DVL different models wins 15.

For 3 objectives, only the SVR and the MO-Linear models achieved good results of hypervolume in DTLZ1, DTLZ3, and DTLZ7, just as they obtained in the regression experiment. For 10 objectives, no model that got the best result in the regression experiment achieved the best HV mean. What draws the most attention is the performance of the SVR in 3 objectives, and the two MLP versions, MLP and MO-MLP, which achieved good results in 3 and 10 objectives. Mainly because although the MLP does not obtain the lowest MSE value, the generated solutions have a good diversity which implies a better HV. Besides that, the architecture of the DVL algorithm allows new training at each iteration with objectives closer to the ideal. As the model is reused during iterations, this task can be seen as transfer learning and many studies have demonstrated the superior performance of neural networks compared to statistical methods in transfer learning.

In comparison with the NSGA-III algorithm, for 3 objectives, in the problems DTLZ1-4 that have simpler Pareto-optimal fronts, all of them triangular with DTLZ1 being plan and DTLZ2, DTLZ3 and DTLZ4 a quarter of a sphere, the DVL algorithm was superior only when there is a higher restriction in the number of evaluations. In a less restrictive scenario, where the number of evaluations is 1500 and 10000, the NSGA-III got better results in these problems. For the problems DTLZ5-7 the scenario is quite different, with the DVL being superior in most cases. These problems are characterized by having Pareto-optimal fronts with different shapes from the previous ones. The case of DTLZ5 and DTLZ6 both have degenerated Pareto-optimal fronts, and DTLZ7 has a disconnected front. These kinds of problems make it difficult for MOEAs to find the ideal frontier.

The results of the experiment with 10 objectives show that the DVL algorithm was superior in the vast majority of cases, except for the case of the DTLZ4 problem with 10000 evaluations, in which the NSGA-III is superior to the DVL with any model. This shows that the DVL can also be used for optimization in problems with a higher number of objectives. We can deduce from the experiments that the DVL has a good performance in different kinds of

problems with different Pareto-optimal shapes, search space sizes, with different local and global optimal, and a different number of objectives. According to the experiments, this last characteristic, the number of objectives, is a key factor for the greater performance of the DVL in comparison to the NSGA-III. The results also show that the strategy used for learning as well as the model is problem-dependent. In different scenarios, different strategies got better results, with the MO-MLP and MLP having the best overall among these strategies.

TABLE II
WINS/TIES/LOSSES OF THE STATISTICAL COMPARISON.

	3 objectives	10 objectives
MO-Linear	0 / 4 / 31	1 / 0 / 34
SVR	4 / 0 / 31	0 / 0 / 35
MO-MLP	1 / 2 / 32	8 / 0 / 27
MO-RFR	0 / 0 / 35	0 / 0 / 35
NSGA-III	14 / 2 / 19	1 / 1 / 33
Linear	1 / 3 / 31	0 / 1 / 34
MLP	5 / 1 / 29	24 / 0 / 11
RFR	4 / 0 / 31	0 / 0 / 35

V. CONCLUSION

In this paper, we showed experimentally that the DVL algorithm and its strategy of inverse modeling can compete with well-known MOEAs of the literature, such as the NSGA-III in the optimization of benchmark problems. Experimental results demonstrate that the inverse model can estimate the Pareto-optimal front of the DTLZ benchmark test suit problems, and also, the DVL algorithm can outperform the NSGA-III algorithm in restrictive scenarios, i.e. with restrictions of the number of evaluations, and in problems with a higher number of objectives. The overall result found is quite promising, the DVL produced a valuable performance against the NSGA-III in many scenarios, even though the DVL algorithm is recent and also it needs greater maturity for optimization tasks.

For future works, we suggest the improvement of the weak points of the DVL algorithm, such as the deterioration of the search performance due to local optima and the shape of the reference points and to test DVL algorithm needs to be experimentally tested in real-world optimization expensive problems.

REFERENCES

- [1] O. Schütze, A. Lara, and C. A. C. Coello, "On the influence of the number of objectives on the hardness of a multiobjective optimization problem," *IEEE Trans. Evolutionary Computation*, vol. 15, no. 4, pp. 444–455, 2011.
- [2] L. Calvet, J. de Armas, D. Masip, and A. A. Juan, "Learnheuristics: hybridizing metaheuristics with machine learning for optimization with dynamic inputs," *Open Mathematics*, vol. 15, no. 1, pp. 261–280, 2017.
- [3] I. Giagkiozis and P. J. Fleming, "Pareto front estimation for decision making," *Evolutionary computation*, vol. 22, no. 4, pp. 651–678, 2014.
- [4] S. Reddy and G. S. Dulikravich, "A self-adapting algorithm for many-objective optimization," *Applied Soft Computing*, vol. 129, p. 109484, 2022.
- [5] "Multi/many-objective evolutionary algorithm assisted by radial basis function models for expensive optimization," *Applied Soft Computing*, vol. 122, p. 108798, 2022.

- [6] S. Liu, H. Wang, W. Yao, and W. Peng, "Surrogate-assisted environmental selection for fast hypervolume-based many-objective optimization," *IEEE Transactions on Evolutionary Computation*, pp. 1–1, 2023.
- [7] M. Santos, J. A. de Oliveira, and A. Britto, "Decision variable learning," in *2019 8th Brazilian Conference on Intelligent Systems (BRACIS)*, pp. 497–502, IEEE, 2019.
- [8] R. Kudikala, I. Giagkiozis, and P. Fleming, "Increasing the density of multi-objective multi-modal solutions using clustering and pareto estimation techniques," in *The 2013 World Congress in Computer Science Computer Engineering and Applied Computing*, 2013.
- [9] Y. Yan, I. Giagkiozis, and P. J. Fleming, "Improved sampling of decision space for pareto estimation," in *Proceedings of the 2015 Annual Conference on Genetic and Evolutionary Computation*, pp. 767–774, 2015.
- [10] R. Cheng, Y. Jin, K. Narukawa, and B. Sendhoff, "A multiobjective evolutionary algorithm using gaussian process-based inverse modeling," *IEEE Transactions on Evolutionary Computation*, vol. 19, no. 6, pp. 838–856, 2015.
- [11] I. Das and J. E. Dennis, "Normal-boundary intersection: A new method for generating the pareto surface in nonlinear multicriteria optimization problems," *SIAM journal on optimization*, vol. 8, no. 3, pp. 631–657, 1998.
- [12] J. J. Sánchez-Medina, M. J. Galán-Moreno, and E. Rubio-Royo, "Traffic signal optimization in "la almazara" district in saragossa under congestion conditions, using genetic algorithms, traffic microsimulation, and cluster computing," *IEEE Transactions on Intelligent Transportation Systems*, vol. 11, no. 1, pp. 132–141, 2009.
- [13] K. Deb, L. Thiele, M. Laumanns, and E. Zitzler, "Scalable test problems for evolutionary multiobjective optimization," in *Evolutionary multiobjective optimization*, pp. 105–145, Springer, 2005.
- [14] J. Derrac, S. García, D. Molina, and F. Herrera, "A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms," *Swarm and Evolutionary Computation*, vol. 1, no. 1, pp. 3–18, 2011.
- [15] K. Deb and H. Jain, "An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting approach, part i: Solving problems with box constraints," *IEEE Transactions on Evolutionary Computation*, vol. 18, no. 4, pp. 577–601, 2014.



Artur Leandro da Costa Oliveira received the B.S. and M.S. degrees in computer science from the Federal University of Sergipe, São Cristóvão, Sergipe, Brazil, in 2011 and 2022 respectively. From 2011 to 2022, he work as a System Analyst. His research interest includes optimization and machine learning areas.



René Gusmão received the Ph.D. degree in computer science from Federal University of Pernambuco, Brazil. He is an adjunct professor of the Federal University of Sergipe. His current research interests include cluster analysis, data mining and computational intelligence.



André Britto received the Ph.D. in computer science department from Federal University of Parana, Parana, Brazil. He is an adjunct professor at the Federal University of Sergipe, Sergipe, Brazil. His main interests are multi-objective optimization and machine learning.