2014 IEEE Congress on Evolutionary Computation (CEC)
July 6-11, 2014, Beijing, China

Differential Evolution Assisted by a Surrogate Model for Bilevel Programming Problems

Jaqueline S. Angelo*, Eduardo Krempser*†, Helio J.C. Barbosa*‡
*Laboratório Nacional de Computação Científica, Petrópolis - RJ, Brazil
†Faculdade de Educação Tecnológica do Estado do Rio de Janeiro (FAETERJ-Petrópolis)
‡Universidade Federal de Juiz de Fora, Juiz de Fora, MG, Brazil
Email: {jsangelo, krempser, hcbm}@lncc.br

***Abstract*—Bilevel programming is used to model decentralized problems involving two levels of decision makers that are hierarchically related. Those problems, which arise in many practical applications, are recognized to be challenging. This paper reports a Differential Evolution (DE) method assisted by a surrogate model to solve bilevel programming problems (BLPs). The method proposed is an extension of a previous one, BlDE, developed by the authors, where two DE methods are used to generate and evolve the upper and the lower level variables. Here, the use of a similarity-based surrogate model, and a different stopping criteria, are proposed in order to reduce the number of function evaluations on both levels of the problem. The numerical results show a significant reduction in the number of function evaluations in the lower level of the problem, as well as some improvement in the upper level.**

I. Introduction

Over the years, a branch of mathematical programming that has become an important area of research is the design and implementation of efficient computational methods to treat the complex problems of bilevel optimization. Bilevel programming problems (BLPs) are considered very difficult to solve, because they contain an optimization problem within the constraints of another optimization problem. Problems of this type are considered more difficult to treat than the classical optimization problems, since, in general, they are non-convex and non-differentiable, even when the functions involved are all linear; in fact in [16], [8] they were proved to be NP-hard.

In the BLP, two decision makers, the leader in the upper level and the follower in the lower level, are hierarchically related, where the leader's decisions affect both the follower's payoff function and allowable actions, and vice-versa. The main feature of such problems is that the decisions at the upper level can influence the decision maker of the lower level, but cannot completely control its actions. In addition, the objective function of one level is usually partially determined by variables controlled by the other level of the hierarchy.

Do to the complexity involved in solving BLPs, intelligent heuristics, such as evolutionary computation, become powerful tools to overcome the many challenges of bilevel programming problems, such as non-convexity and non-differentiability, large number of variables and/or constraints, mixed types of design variables and non-unique optimal solution for the follower's problem. However, heuristic methods often require a large number of fitness and constraint evaluations. This becomes a serious drawback in situations where expensive simulations are required.

Since in this paper we are interested in developing an evolutionary method capable to solve bilevel problems with complex simulation models, which usually require a large computational time to be computed, we propose the use of a surrogate model (or metamodel) and a different stopping criteria, to replace the lower level optimization by a relatively inexpensive approximation of the lower level function, so as to reduce the number of calls to the (expensive) objective function evaluator.

In this paper, a simple similarity-based surrogate model, and a different stopping criterion are applied to the BlDE algorithm, previously proposed in [4], in order to reduce the number of upper and lower level function evaluations. The method uses two nested Differential Evolution algorithms, each one responsible for optimizing one level of the problem. Firstly, the proposed method is tested on a variety of test problems taken from the literature, which include linear, non-linear, constrained and unconstrained optimization problems. Secondly, the well known SMD test-problems [26] are used to evaluate the proposed method.

In the next section, we present the formulation of a general bilevel optimization problem and describe the notion of optimal solution for this problem. In Section III the Differential Evolution algorithm is presented where the different variants used in the bilevel method proposed are described. In the next section we present a description of the surrogate model used to assist the DE. Section V describes the proposed bilevel methodology that utilizes a surrogate model within the two nested DE algorithms. The standard test problems and the SMD problems, used to evaluate the proposed method, are described in Section VI. Thereafter, the computational results are discussed. Finally, the conclusions are presented in Section VIII.

II. Bilevel Programming

In bilevel programming problems, two decision makers, the leader (L) in the upper level and the follower (F) in the lower level, are hierarchically related. The main characteristic of BLPs is that the leader's decisions affect both the follower's payoff function and its allowable actions, and vice-versa. Each decision maker has control over a set of variables, seeking to optimize his own objective function. The leader has control over the x variables, and makes his decision first, fixing x, while the follower has control over the y variables. Reacting to the decision of the leader, the y variables are set in response to the given x.

978-1-4799-1488-3/14/$31.00 ©2014 IEEE

A bilevel programming problem can be written as:

$$
\begin{array}{lll}
(L) & \min\limits_{x \in X} & f_1(x, y(x)) \\
& \text{subject to} & g_1(x, y(x)) \leq 0 \\
& (F) & y(x) \in R(x) := \arg\min\limits_{y \in Y} f_2(x, y) \\
& & \text{subject to} \quad g_2(x, y) \leq 0
\end{array}
\tag{1}
$$

where $f_1(x, y(x))$ and $f_2(x, y)$ are the upper and lower level objective functions, respectively, with $g_1(x, y(x))$ and $g_2(x, y)$ being their respective constraints. $x \in X \subset R^{n_1}$ are the upper level variables and $y \in Y \subset R^{n_2}$ are the lower level variables. The reaction set of the follower $R(x)$ defines the follower's response given a fixed x by the leader. To ensure that (1) is well posed it is common to assume that for all decisions taken by the leader, the follower has some room to respond, i.e., $R(x) \neq \emptyset$.

The feasible set of the bilevel problem (1) is

$$\Omega := \{(x, y) : x \in X, y \in Y, g_1(x, y(x)) \leq 0, g_2(x, y) \leq 0\}$$

and the feasible set of the follower, for each $x \in X$, is

$$\Omega_y := \{y \in Y : g_2(x, y) \leq 0\}$$

A minimizing solution $y(x)$, for the follower's problem, in response to a given x fixed by the leader, satisfies the following relation [25]:

$$f_2(x, y(x)) \leq f_2(x, y) \quad \forall y \in \Omega_y$$

For such $y(x)$, if it exists $x^* \in X$, such that

$$f_1(x^*, y(x^*)) \leq f_1(x, y(x)) \quad \forall x \in \Omega$$

then the solution (x^*, y^*), where $y^* = y(x^*)$, is the optimal solution for the bilevel problem with y^* being the optimal solution for the follower's problem in response to x^*.

A. Difficulties in solving BLPs

One difficulty that arises in solving a BLP is that, if $R(x)$ is not single-valued for all possible x, the leader may not achieve his minimum payoff, since the follower has multiple minimum solutions to choose from. In this case, there is no guarantee that the follower's choice is the best for the leader, leading to sub-optimal solutions in the leader's problem.

To overcome this situation at least two approaches can be considered; the optimistic one and the pessimistic one. In the optimistic case, the leader assumes that the follower is willing to support him, i.e., that the follower will select a solution $y(x) \in R(x)$ which is the best from the leader's point-of-view. This results in the so-called *optimistic* or *weak bilevel problem* [12]:

$$
\begin{array}{lll}
(L) & \min\limits_{x \in X} \min\limits_{y \in R(x)} & f_1(x, y(x)) \\
& \text{subject to} & g_1(x, y(x)) \leq 0 \\
& (F) & y(x) \in R(x) := \arg\min\limits_{y \in Y} f_2(x, y) \\
& & \text{subject to} \quad g_2(x, y) \leq 0
\end{array}
\tag{2}
$$

On the other hand, in the pessimistic case, the leader protects himself against the worst possible situation, leading to the so-called *pessimistic* or *strong bilevel problem* [12]:

$$
\begin{array}{lll}
(L) & \min\limits_{x \in X} \max\limits_{y \in R(x)} & f_1(x, y(x)) \\
& \text{subject to} & g_1(x, y(x)) \leq 0 \\
& (F) & y(x) \in R(x) := \arg\min\limits_{y \in Y} f_2(x, y) \\
& & \text{subject to} \quad g_2(x, y) \leq 0
\end{array}
\tag{3}
$$

Another challenge lies in the fact that unless a solution is optimal for the lower level problem, it cannot be feasible for the overall problem. This suggests that approximate methods could not be used to solve the lower level problem, as they are not guaranteed to reach the optimal solution. However, the complexity of many bilevel applications makes the use of exact methods impractical.

III. Differential Evolution

Differential Evolution (DE) is a stochastic population-based algorithm for global optimization, considered very simple and easy to use because it requires very few control parameters. The basic operation of DE is to perturb the current population members with scaled differences of distinct randomly selected population members. The variants (strategies) of DE are determined by the number of differences applied, the way in which the individuals are selected, and the distribution of recombination. The DE performance depends on the variant chosen, and here two DE variants proposed in [20] are applied and evaluated:

DE/best/1/bin: The new individual generated uses the best individual in the population $x_{best,j,G}$ as base vector in the mutation, and r_1 and r_2 indicate randomly selected individuals, leading to

$$x_{best,j,G} + F.(x_{r_1,j,G} - x_{r_2,j,G}) \tag{4}$$

DE/target-to-best/1/bin: This variant uses the best individual of the population and the target individual (the one that will be used in the comparison after the mutation, also called current individual), to generate a new individual, leading to

$$x_{i,j,G} + F.(x_{best,j,G} - x_{i,j,G}) + F.(x_{r_1,j,G} - x_{r_2,j,G}) \tag{5}$$

In addition, a crossover operation is performed, using the parameter CR. Also, for each design variable, lower and upper bounds are usually applied. Whenever a given component x_i of a candidate solution x is generated outside its prescribed range, a standard projection operation is performed:

$$\text{If } x_{i,j} > x_j^U \text{ then } x_{i,j} = x_j^U; \quad \text{if } x_{i,j} < x_j^L \text{ then } x_{i,j} = x_j^L.$$

IV. DE Assisted by a Surrogate Model

Replacing the original evaluation function (a complex computer simulation) by a substantially less expensive approximation is known as surrogate modeling, or metamodeling. This idea appeared early in the evolutionary computation literature [15] and many possibilities are available today (see [14] for a survey). In the context of Differential Evolution, many surrogate models were already proposed such as artificial neural networks [27], radial basis function networks [19] and nearest neighbors techniques [17].

Similarity-Based Surrogate Models (SBSM) store their inputs and defer processing until a prediction of the fitness value of a new candidate solution is requested. Thus, SBSM can be classified as "lazy" learners or memory-based learners [1]. Our proposal is to apply a surrogate model, based on nearest neighbors techniques, aiming at reducing the number of objective function evaluations. The k-Nearest Neighbors (k-NN) [23] was used, in which the k nearest candidate solutions are selected.

In the BlDE method [4], for each x fixed, a DE method was performed to obtain the y values. However, this process required a large number of lower level function evaluations. Therefore, we propose to replace the DE follower process by a k-NN approximation. So, when the x values are selected we have two situations: (i) the DE follower process is applied, and the y values are obtained or (ii) an approximated method is applied to calculate the y values, using equation (6).

When the follower DE is applied, the x and y values are stored in the archive $\mathcal{D}$. When the approximation method is selected to be applied, the y values are calculated based in the x values and the archive $\mathcal{D}$.

Given a candidate solution x and the archive $\mathcal{D} = \{(x_i, y(x_i)), i = 1, \dots, \eta\}$, containing the solutions evaluated by the follower DE, the following approximation is considered:

$$y(x) \approx \widehat{y}(x) = \frac{\sum_{j=1}^{|\mathcal{N}|} s(x, x_j^{\mathcal{N}})^p y(x_j^{\mathcal{N}})}{\sum_{j=1}^{|\mathcal{N}|} s(x, x_j^{\mathcal{N}})^p} \tag{6}$$

where η is the size of the archive $\mathcal{D}$, $|\mathcal{N}|$ denotes the cardinality of the set $\mathcal{N}$ composed by the k elements in the set $\mathcal{D}$ most similar to x. The $x_j^{\mathcal{N}} \in \mathcal{N}$ are the nearest neighbors of x, $s(x, x_j^{\mathcal{N}})$ is a similarity measure between x and $x_j^{\mathcal{N}}$, and p is set to 2. Here, $s(x, x_j^{\mathcal{N}}) = [d_E(x, x_j^{\mathcal{N}})]^{-1}$, where $d_E(x, x_j^{\mathcal{N}})$ is the Euclidean distance between x and $x_j^{\mathcal{N}}$. If $x = x_i$ for some $x_i \in \mathcal{D}$ then $\widehat{y}(x) = y(x_i)$.

V. The Proposed Methodology

Algorithms 1 and 2 describe the upper –leader– and lower –follower– level optimization of the proposed method. The main steps of the algorithm are summarized as follows:

Step 0: Initialization. The algorithm starts with a population, of size POP$_u$, of vectors containing the upper level variables $x \in R^{n_1}$. The upper level variables are initialized with random values and the lower level variables are determined by executing the lower level procedure (Algorithm 2), which generates the vector $y \in R^{n_2}$ of lower level variables.

Step 1: Upper level procedure. Following the basic DE algorithm described in Algorithm 1, the upper level individuals are mutated and recombined.

Step 2: Evaluation of each upper level individual. To evaluate the individuals in the upper level, where fitness is assigned based on the upper level function and constraints, the lower level procedure is performed. The solution returned, that is, the best individual obtained in the lower level procedure, is used to evaluate the upper level individual.

Step 3: Lower level procedure. In order to evaluate the lower level problem, two different procedures can be applied:

Step 3.1: Evolutionary model. For fixed upper level variables, a new DE algorithm is executed, as described in Algorithm 2. The individuals are evaluated based on the lower level function and constraints. Finally, the procedure returns the best value of the lower level problem. After this process the x variables and its associated y are stored in the archive $\mathcal{D}$.

Step 3.2: Surrogate model. For fixed upper level variables, the equation (6) and the archive $\mathcal{D}$ are used to obtain the associated y variables.

Algorithm 1: Algorithm DE Leader.

input : POP$_u$ (population size), F (mutation scaling), CR (crossover rate)

G $= 0$;
`CreateRandomInitialPopulation(`POP$_u$`);`
for $i \leftarrow 1$ **to** POP$_u$ **do**
  $\overrightarrow{y}$ = `DEFollower(`*POP$_l$, F, CR,* $\overrightarrow{x}_{i,G}$`);`
  `Evaluate` f_1`(`$\overrightarrow{x}_{i,\mathsf{G}}$, $\overrightarrow{y}$`) ;   /*` $\overrightarrow{x}_{i,\mathsf{G}}$ `is an individual in the population */`
  `InsertDatabase(`$\overrightarrow{x}_{i,G}$, $\overrightarrow{y}$`)`
while *termination criteria not satisfied* **do**
  G ++;
  for $i \leftarrow 1$ **to** POP$_u$ **do**
    `SelectRandomly(`r_1, r_2, r_3`);`
    `/*` $r_1 \neq r_2 \neq r_3 \neq i$ `*/`
    jRand ←`RandInt(`*1,* n_1`)`
    for $j \leftarrow 1$ **to** n_1 **do**
      if `Rand(0,1)` $<$ CR *or* $j = jRand$ **then**
        $u_{i,j,G+1}$ = equation (4) or (5)
      else
        $u_{i,j,G+1}$ = $x_{i,j,G}$;
    if `Rand(0,1)` $\leq \beta$ *and* $G \geq \gamma$ **then**
      $\overrightarrow{y}$ = `AproximatedFollower(`*POP$_l$,* $\overrightarrow{u}_{i,G+1}$`);`
    else
      $\overrightarrow{y}$ = `DEFollower(`*POP$_l$, F, CR,* $\overrightarrow{u}_{i,G+1}$`);`
      `InsertDatabase(`$\overrightarrow{u}_{i,G+1}$, $\overrightarrow{y}$`)`
    if $f_1(\overrightarrow{u}_{i,G+1}, \overrightarrow{y}) \leq f_1(\overrightarrow{x}_{i,G}, \overrightarrow{y})$ **then**
      $\overrightarrow{x}_{i,G+1}$ = $\overrightarrow{u}_{i,G+1}$;
    else
      $\overrightarrow{x}_{i,G+1}$ = $\overrightarrow{x}_{i,G}$;

A. Constraint handling

The upper and lower level constraints of the bilevel problems are handled by the method proposed in [11], which enforces the following criteria: (i) any feasible solution is preferred to any infeasible solution; (ii) among two feasible solutions, the one having better objective function value is preferred, and (ii) among two infeasible solutions, the one having smaller constraint violation is preferred.

B. Termination criteria

The algorithm uses a variance based termination criterion in each level of the bilevel optimization [26]. At the upper level, when the value of α_u, described in (7), becomes less than α_u^{stop}, the upper level algorithm terminates.

$$\alpha_u = \sum_{i=1}^{n_1} \frac{\sigma^2(x_i^t)}{\sigma^2(x_i^{initial})} \tag{7}$$

where n_1 is the number of upper level variables, x_i^t are the upper level variables in generation t and $x_i^{initial}$ are the upper level variables in the initial population, with $i \in \{1, ..., n_1\}$.

Algorithm 2: Algorithm DE Follower.

input : POP_l (follower population size), F (mutation scaling), CR (crossover rate), $\overrightarrow{\mathsf{V}}$ (leader variables)

$\mathsf{G} = 0$;
`CreateRandomInitialPopulation(`POP_l`);`
for $i \leftarrow 1$ **to** POP_l **do**
    `Evaluate` $f_2(\overrightarrow{\mathsf{V}}, \overrightarrow{x}_{i,\mathsf{G}})$; `/*` $\overrightarrow{x}_{i,\mathsf{G}}$ `is an individual in the population */`
while *termination criteria not satisfied* **do**
    G ++;
    for $i \leftarrow 1$ **to** POP_l **do**
        `SelectRandomly(`r_1, r_2, r_3`) ; /*` $r_1 \neq r_2 \neq r_3 \neq i$ `*/`
        $jRand \leftarrow$`RandInt(`*1*, n_2`)`
        for $j \leftarrow 1$ **to** n_2 **do**
            if `Rand(0,1)` $< \mathsf{CR}$ *or* $j = jRand$ **then**
                $u_{i,j,G+1}$ = equation (4) or (5)
            else
                $u_{i,j,G+1} = x_{i,j,G}$;
        if $f_2(\overrightarrow{\mathsf{V}}, \overrightarrow{u}_{i,G+1}) \leq f_2(\overrightarrow{\mathsf{V}}, \overrightarrow{x}_{i,G})$ **then**
            $\overrightarrow{x}_{i,G+1} = \overrightarrow{u}_{i,G+1}$;
        else
            $\overrightarrow{x}_{i,G+1} = \overrightarrow{x}_{i,G}$;
return `SelectBestIndividual`

For the lower level, when the value of α_l, described in (8), becomes less than α_l^{stop}, the lower level algorithm terminates.

$$\alpha_l = \sum_{i=1}^{n_2} \frac{\sigma^2(y_i^t)}{\sigma^2(y_i^{initial})} \tag{8}$$

where n_2 is the number of lower level variables, y_i^t are the lower level variables in generation t and $x_i^{initial}$ are the lower level variables in the initial population, with $i \in \{1, ..., n_2\}$.

VI. Test Problems

The results obtained by the proposed method are analyzed using 25 test-problems divided in two groups. Due to the lack of space, the description of the problems is omitted (they are all available in [4], except for problem 19, available in [9]).

A. Standard test problems

First, the proposed method is applied on a variety of test problems from the literature [25], [10], [6], [2], [7], [3], [22], [13], [24], [5], [18], [21], [9]. Those problems include linear, non-linear, constrained, and unconstrained optimization problems, most of them with 2 or 4 decision variables, and one of them with 8 decision variables.

B. SMD test-problems

The second part of the experiments consists in solving the unconstrained test-collection (SMD1 to SMD6) proposed in [26]. Those problems aim to induce difficulties at both levels of the BLP, independently and collectively, such that the performance of the algorithms could be better evaluated when handling the two levels.

For those problems the instances considered have 10 decision variables and correspond to setting $p = 3$, $q = 3$, and $r = 2$ for problems SMD1 to SMD5, and $p = 3$, $q = 1$, $r = 2$, and $s = 2$ for problem SMD6.

VII. Computational Results

The algorithm proposed was first tested in 19 test problems taken from different sources in the literature. In the second part of the tests the performance of the proposed method was analyzed using the SMD test-problems. As described in Section III two variants of DE were considered, DE/target-to-rand/1/bin and DE/best/1/bin.

The experiments analyze the results obtained by the proposed algorithm when a surrogate model is used to replace the lower level optimization. The proposed method, with different probabilities β of using the surrogate model, is analyzed from the point of view of the quality of the solutions found and the number of exact function evaluations saved.

A. Parameter setting

The proposed method was executed 30 times for each test problem, using the following parameter setting:

- F: the scale factor –mutation rate– is set to 0.8.
- CR: the crossover probability is set to 0.9.
- POP_u and POP_l: the upper and the lower level population size are both set to 30.
- α_u^{stop} and α_l^{stop}: the accuracy in both termination criteria is set to 0.00001.
- β: the probability of using the metamodel varies among 0 (no use), 0.3, 0.5, and 0.8.
- k: the number of nearest candidate solutions selected to calculate the lower level variables via the metamodel is set to 2.
- γ: the initial number of generations in which the surrogate model is not applied is set to 1.

B. Results for the standard test problems

Because of the diversity of the 19 test problems and the very aggressive search of DE variant DE/best/1/bin, the proposed method, using this variant on both levels, did not perform well on those problems. Thereby, the results presented on Tables I to IV correspond only to the use of the DE/target-to-rand/1/bin variant on both levels of the optimization.

Tables I and II describe the median and mean objective functions values of the upper (UL) and lower (LL) level problems, where BKS indicates the best known solutions. Tables III and IV present the median and mean values of the number of function evaluations (FE) for the upper and lower level problems, and the last column indicates the percentage of savings on the number of lower level function evaluations (%LLSav).

C. Results for the SMD test-problems

Table V describes the median and mean values of the upper (UL) and lower (LL) level objective functions, where "target" means the variant DE/target-to-rand/1/bin and "best" means the variant DE/best/1/bin. For those problems the best known solution for the upper and lower problems are both zero. Table VI presents the median and mean values of number of function evaluations (FE) for the upper and lower level problems.

D. Discussions

From Table I and II it is possible to observe that the proposed method using no metamodel ($\beta = 0$) was capable to reach, or get very close to, the best known solutions in all problems tested. However, when the probability of using the metamodel increases, for some problems, the method cannot reach the best known solutions. It seems that when $\beta \geq 0.5$ the solutions deviate from the expected results.

When the surrogate model is not able to obtain the expected values, the optimization process can be directed to false optimal solutions (minimum solutions of the approximated function), leading to poor quality solutions. Furthermore, in some cases, the surrogate method can even slow down the convergence of the upper level optimization.

Tables III and IV show that the number of lower level function evaluations is significantly reduced (except for problem 16) as the percentage of using the surrogate model increases, as indicated by the percentage of savings on the number of lower level function evaluations. We can highlight that although the metamodel has been used to reduce the number of function evaluations of the lower level, for problems 7, 11, 15, and 17 the number of upper level function evaluations also decreased as the use of the metamodel increased.

For the SMD problems, in both variants, the proposed method efficiently solve all problems when no metamodel was used. In fact, for problems SMD1 and SMD3, with a high probability of using the metamodel ($\beta = 0.8$), the method still solves efficiently these problems. However, as happened with the standard test problems, for problems SMD2, SMD4, SMD5, and SMD6, when $\beta \geq 0.5$ the solutions deviate from the expected results.

From Table VI it is possible to observe a significant reduction on the number of lower level function evaluations for both DE variants in all problems tested, reaching a reduction of over 75% in problems SMD1 and SMD3, when $\beta = 0.8$. One can observe that the variant DE/best/1/bin presented a reduced number of function evaluations when compared with the variant DE/target-to-best/1/bin for all values of β in all SMD test-problems.

VIII. Conclusion

In this paper we proposed to improve the BlDE algorithm, previously developed by the authors [4], in order to reduce the number of objective function evaluations in bilevel optimization problems. The new method implements a nested technique where each DE algorithm is responsible for optimizing one level of the bilevel problem, uses a different termination criterion, and is equipped with a surrogate model in the lower level optimization.

TABLE I. Median and mean values of the upper (UL) and lower (LL) level objective functions (Functions 1 – 12)

Problem 1					
β	UL Med.	UL Mean	LL Med.	LL Mean	
0.8	99.75	99.18	0.002961	0.1089	UL BKS
0.5	99.99	99.57	0.0001027	0.07055	100
0.3	100	99.73	1.008e-05	0.05117	LL BKS
0.0	100	100	2.212e-06	4.061e-06	0
Problem 2					
β	UL Med.	UL Mean	LL Med.	LL Mean	
0.8	224.9	223	99.13	94.23	UL BKS
0.5	225.1	224.5	99.77	99.16	225
0.3	225.1	225	99.76	99.36	LL BKS
0.0	225.1	225.1	99.83	99.79	100
Problem 3					
β	UL Med.	UL Mean	LL Med.	LL Mean	
0.8	28.43	28.55	-3.154	-5.816	UL BKS
0.5	28.76	28.72	-3.156	-3.135	29.2
0.3	28.86	28.39	-3.182	-3.276	LL BKS
0.0	28.91	28.84	-3.174	-3.16	-3.2
Problem 4					
β	UL Med.	UL Mean	LL Med.	LL Mean	
0.8	3.327	3.13	2.461	2.268	UL BKS
0.5	3.31	3.386	2.715	2.322	3.25
0.3	3.251	3.337	3.587	2.824	LL BKS
0.0	3.247	3.245	3.945	3.936	4
Problem 5					
β	UL Med.	UL Mean	LL Med.	LL Mean	
0.8	0	-1.661	200	192.2	UL BKS
0.5	0	-0.6545	200	198.1	0
0.3	0	-0.5524	200	198.7	LL BKS
0.0	0	-1.409e-06	200	193.3	200
Problem 6					
β	UL Med.	UL Mean	LL Med.	LL Mean	
0.8	16.32	12.43	0.6178	-1.548	UL BKS
0.5	17	14.77	0.9964	-0.05756	17
0.3	17	15.14	0.9989	0.06603	LL BKS
0.0	17	17	0.9989	0.9925	1
Problem 7					
β	UL Med.	UL Mean	LL Med.	LL Mean	
0.8	-12.82	-12.8	-0.9483	-0.8823	UL BKS
0.5	-12.83	-12.84	-0.933	-0.9066	-12.679
0.3	-12.83	-12.83	-0.9578	-0.9172	LL BKS
0.0	-12.82	-12.83	-0.9686	-0.951	-1.015
Problem 8					
β	UL Med.	UL Mean	LL Med.	LL Mean	
0.8	48.84	47.39	-16.89	-15.85	UP BKS
0.5	48.96	48.59	-16.98	-16.71	49
0.3	48.97	48.96	-16.98	-16.97	LL BKS
0.0	48.96	48.95	-16.98	-16.97	-17
Problem 9					
β	UL Med.	UL Mean	LL Med.	LL Mean	
0.8	-1.544	-1.658	7.948	8.555	UL BKS
0.5	-1.416	-1.51	7.627	8.031	-1.407
0.3	-1.405	-1.422	7.61	7.727	LL BKS
0.0	-1.407	-1.405	7.616	7.606	7.61
Problem 10					
β	UL Med.	UL Mean	LL Med.	LL Mean	
0.8	-1.025	-1.024	0.00113	0.01522	UL BKS
0.5	-1.024	-1.035	0.001833	0.00852	-1
0.3	-1.017	-1.037	0.0006187	0.02157	LL BKS
0.0	-1.015	-1.015	0.000304	0.0003688	0
Problem 11					
β	UL Med.	UL Mean	LL Med.	LL Mean	
0.8	2049	1913	85.91	3937	UL BKS
0.5	2099	1963	155.6	4077	2250
0.3	2210	2119	156.9	472	LL BKS
0.0	2248	2248	197.3	192.5	197.75
Problem 12					
β	UL Med.	UL Mean	LL Med.	LL Mean	
0.8	-12.02	-12.77	4	4.069	UL BKS
0.5	-12.02	-12.54	4	4.058	-12
0.3	-11.99	-12.4	3.997	4.038	LL BKS
0.0	-11.99	-11.98	3.997	3.993	4

TABLE II. MEDIAN AND MEAN VALUES OF THE UPPER (UL) AND LOWER (LL) LEVEL OBJECTIVE FUNCTIONS (FUNCTIONS 13 – 19)

Problem 13

β	UL Med.	UL Mean	LL Med.	LL Mean	
0.8	3.117	3.262	-6.696	-7.312	UL BKS
0.5	3.117	3.117	-6.696	-6.699	3.111
0.3	3.113	3.114	-6.686	-6.702	LL BKS
0.0	3.113	3.117	-6.682	-6.696	-6.662

Problem 14

β	UL Med.	UL Mean	LL Med.	LL Mean	
0.8	1	0.839	0	175.2	UL BKS
0.5	1	0.9553	0	90.97	1
0.3	1	0.9996	0	27.77	LL BKS
0.0	1	1	0	0	0

Problem 15

β	UL Med.	UL Mean	LL Med.	LL Mean	
0.8	951.3	760.5	1	1	UL BKS
0.5	1000	854.4	1	1	1000
0.3	1000	881.1	1	1	LL BKS
0.0	1000	885.2	1	1	1

Problem 16

β	UL Med.	UL Mean	LL Med.	LL Mean	
0.8	4.766	4.581	3.845	4.025	UL BKS
0.5	4.805	4.738	3.911	3.904	5
0.3	4.957	4.88	4.168	4.075	LL BKS
0.0	4.997	4.997	4.025	4.033	4

Problem 17

β	UL Med.	UL Mean	LL Med.	LL Mean	
0.8	9	9	3.974e-13	1.804e-11	UL BKS
0.5	9	9	7.861e-14	4.893e-12	9
0.3	9	9	5.652e-14	3.538e-13	LL BKS
0.0	9	9	1.36e-14	1.813e-13	0

Problem 18

β	UL Med.	UL Mean	LL Med.	LL Mean	
0.8	84.78	81.76	-50.07	-49	UL BKS
0.5	84.94	83.55	-50.13	-49.64	85.09
0.3	85.01	84.93	-50.15	-50.12	LL BKS
0.0	85.02	84.98	-50.15	-50.14	-50.181

Problem 19

β	UL Med.	UL Mean	LL Med.	LL Mean	
0.8	0.1522	0.1617	0.5003	0.5755	UL BKS
0.5	0.1851	0.1908	0.4238	0.5248	0.081
0.3	0.1747	0.1833	0.4197	0.5119	LL BKS
0.0	0.1713	0.1801	0.5467	0.5507	0.666

The experiments showed that the proposed method was capable to efficiently solve all problems tested when the probability of using the surrogate model is about 30% and 50%, providing a significant reduction on the number of lower level function evaluations. The results also indicate that the surrogate model used may be too simple to efficiently solve the variety of test problems considered. When a high probability ($\beta > 0.5$) of using the surrogate model is applied, in some cases the method generated poor quality solutions, and the convergence of the upper level was compromised.

In this way, as future work, it is intended to study new surrogate models for both levels of bilevel optimization problems so as to significantly reduce the number of upper and lower level objective function evaluations without compromising the quality of the final solutions.

ACKNOWLEDGMENT

The authors would like to thank the support from CNPq (grants 141519/2010-0, 140785/2009-4, 308317/2009-2) and Fundação Flora (grant 009/2013/FIOCRUZ/PROBIOII).

REFERENCES

[1] David W. Aha. Editorial. *Artif. Intell. Rev.*, 11(1-5):1–6, 1997. Special issue on lazy learning.

TABLE III. MEDIAN AND MEAN VALUES OF FUNCTION EVALUATIONS (FE) FOR THE UPPER AND LOWER LEVEL PROBLEMS (FUNCTIONS 1 – 11)

Problem 1

β	ULFE Med.	ULFE Mean	LLFE Med.	LLFE Mean	%LLSav.
0.8	840	1677	140200	246200	63.076
0.5	750	1615	259200	512000	31.736
0.3	690	1035	269500	433400	29.023
0.0	631	634.5	379700	379800	-

Problem 2

β	ULFE Med.	ULFE Mean	LLFE Med.	LLFE Mean	%LLSav.
0.8	4425	4039	1470000	1541000	52.396
0.5	1966	2211	1756000	1940000	43.135
0.3	1834	2268	2334000	2776000	24.417
0.0	1818	1818	3088000	3094000	-

Problem 3

β	ULFE Med.	ULFE Mean	LLFE Med.	LLFE Mean	%LLSav.
0.8	1296	1403	768700	816500	66.535
0.5	1040	1111	1465000	1430000	36.221
0.3	1024	1014	1819000	1838000	20.810
0.0	963.5	974.3	2297000	2391000	-

Problem 4

β	ULFE Med.	ULFE Mean	LLFE Med.	LLFE Mean	%LLSav.
0.8	1264	1435	1065000	1070000	61.160
0.5	875	890.7	1646000	1651000	39.971
0.3	822.5	837.2	2013000	2072000	26.586
0.0	758.5	751.2	2742000	2715000	-

Problem 5

β	ULFE Med.	ULFE Mean	LLFE Med.	LLFE Mean	%LLSav.
0.8	6040	5832	613400	605100	77.597
0.5	6036	6037	1471000	1456000	46.275
0.3	6032	6035	1954000	1973000	28.634
0.0	6032	6033	2738000	2926000	-

Problem 6

β	ULFE Med.	ULFE Mean	LLFE Med.	LLFE Mean	%LLSav.
0.8	6030	4987	495100	438800	80.643
0.5	6031	5852	1271000	1245000	49.901
0.3	6031	6032	1766000	1776000	30.390
0.0	6030	5844	2537000	2442000	-

Problem 7

β	ULFE Med.	ULFE Mean	LLFE Med.	LLFE Mean	%LLSav.
0.8	1812	2805	507900	649700	88,643
0.5	2908	3602	1546000	1924000	65.429
0.3	4602	3792	3242000	2808000	27.504
0.0	4306	3958	4472000	4122000	-

Problem 8

β	ULFE Med.	ULFE Mean	LLFE Med.	LLFE Mean	%LLSav.
0.8	900.5	994	122700	130500	61.811
0.5	783	806.3	223900	219400	30.314
0.3	734.5	736.7	256300	262900	20.230
0.0	653.5	661.1	321300	324200	-

Problem 9

β	ULFE Med.	ULFE Mean	LLFE Med.	LLFE Mean	%LLSav.
0.8	774	2453	112800	286600	57.737
0.5	678	1813	192300	459000	27.951
0.3	568.5	1304	215900	454700	19.108
0.0	553	559	266900	272000	-

Problem 10

β	ULFE Med.	ULFE Mean	LLFE Med.	LLFE Mean	%LLSav.
0.8	902	1660	196300	291900	66.496
0.5	830.5	1289	383300	538300	34.579
0.3	789	1154	478500	632700	18.331
0.0	728	739	585900	595400	-

Problem 11

β	ULFE Med.	ULFE Mean	LLFE Med.	LLFE Mean	%LLSav.
0.8	6030	5848	556300	541900	81.052
0.5	6032	6033	1500000	1485000	48.910
0.3	6032	6016	2089000	2089000	28.849
0.0	6033	6033	2936000	2945000	-

TABLE IV. MEDIAN AND MEAN VALUES OF FUNCTION EVALUATIONS (FE) FOR THE UPPER AND LOWER LEVEL PROBLEMS (FUNCTIONS 12 – 19)

Problem 12

β	ULFE Med.	ULFE Mean	LLFE Med.	LLFE Mean	%LLSav.
0.8	941.5	2719	279900	372100	55.705
0.5	670	1892	399000	626200	36.857
0.3	604	1513	494300	733500	21.776
0.0	583	581.2	631900	642600	-

Problem 13

β	ULFE Med.	ULFE Mean	LLFE Med.	LLFE Mean	%LLSav.
0.8	690	690.7	86200	88980	65.589
0.5	604	613.6	152000	158200	39.321
0.3	573	581.1	189400	188500	24.391
0.0	561	561.1	250500	250300	-

Problem 14

β	ULFE Med.	ULFE Mean	LLFE Med.	LLFE Mean	%LLSav.
0.8	453	1934	70580	162400	57.172
0.5	424.5	991.9	105700	205500	35.862
0.3	439	621.9	129300	175700	21.541
0.0	420	433	164800	169500	-

Problem 15

β	ULFE Med.	ULFE Mean	LLFE Med.	LLFE Mean	%LLSav.
0.8	6094	5555	2604000	2518000	88.795
0.5	6462	5849	9650000	8123000	58.477
0.3	6719	6063	14780000	12610000	36.403
0.0	7057	6399	23240000	20090000	-

Problem 16

β	ULFE Med.	ULFE Mean	LLFE Med.	LLFE Mean	%LLSav.
0.8	6030	5681	821800	801800	-70.533
0.5	6030	6030	1864000	1874000	-286.802
0.3	6030	5665	2530000	2408000	-425.005
0.0	615	2119	481900	1339000	-

Problem 17

β	ULFE Med.	ULFE Mean	LLFE Med.	LLFE Mean	%LLSav.
0.8	480	490	65940	61410	70.109
0.5	480	494	117800	121200	46.600
0.3	480	478	156300	154800	29.148
0.0	510	491	220600	213400	-

Problem 18

β	ULFE Med.	ULFE Mean	LLFE Med.	LLFE Mean	%LLSav.
0.8	1114	1588	148400	191400	59.298
0.5	874.5	1242	226200	332700	37.050
0.3	785	1012	277500	351900	23.889
0.0	741.5	747.4	364600	369600	-

Problem 19

β	ULFE Med.	ULFE Mean	LLFE Med.	LLFE Mean	%LLSav.
0.8	5388	4464	5318000	5404000	8.421
0.5	1635	2291	4829000	6748000	16.842
0.3	1461	2171	5336000	8668000	8.111
0.0	1116	1391	5807000	7449000	-

TABLE V. MEDIAN AND MEAN VALUES OF THE UPPER (UL) AND LOWER (LL) LEVEL OBJECTIVE FUNCTIONS (SMD)

SMD 1

β	Variant	UL Med.	UL Mean	LL Med.	LL Mean
0.8	target	5.018e-05	5.096e-05	3.396e-05	3.382e-05
0.5	target	4.157e-05	4.456e-05	2.207e-05	2.861e-05
0.3	target	3.754e-05	4.068e-05	2.09e-05	2.472e-05
0.0	target	4.209e-05	4.34e-05	2.229e-05	2.259e-05
0.8	best	4.849e-05	7.222e-05	2.685e-05	4.241e-05
0.5	best	4.506e-05	4.638e-05	2.699e-05	3.093e-05
0.3	best	3.439e-05	3.507e-05	1.83e-05	2.291e-05
0.0	best	3.786e-05	3.993e-05	1.833e-05	2.172e-05

SMD 2

β	Variant	UL Med.	UL Mean	LL Med.	LL Mean
0.8	target	-0.7954	-2.235	3.522	7.213
0.5	target	-0.08149	-0.3615	0.3892	2.136
0.3	target	-2.872e-05	-0.1987	0.0002816	1.029
0.0	target	9.218e-06	1.094e-05	7.948e-06	8.786e-06
0.8	best	-1.854	-2.494	4.703	8.028
0.5	best	-0.0008475	-0.4245	0.007381	3.17
0.3	best	3.01e-06	-0.1375	3.595e-05	1.485
0.0	best	1.175e-05	1.159e-05	7.196e-06	7.062e-06

SMD 3

β	Variant	UL Med.	UL Mean	LL Med.	LL Mean
0.8	target	3.214e-05	3.543e-05	1.412e-05	2e-05
0.5	target	3.411e-05	3.554e-05	1.954e-05	2.282e-05
0.3	target	3.885e-05	3.992e-05	1.905e-05	2.354e-05
0.0	target	3.017e-05	3.244e-05	1.418e-05	1.828e-05
0.8	best	3.725e-05	3.769e-05	1.863e-05	2.419e-05
0.5	best	3.547e-05	3.857e-05	2.037e-05	2.402e-05
0.3	best	3.816e-05	3.898e-05	1.84e-05	2.037e-05
0.0	best	3.713e-05	4.048e-05	1.501e-05	2.061e-05

SMD 4

β	Variant	UL Med.	UL Mean	LL Med.	LL Mean
0.8	target	-0.2956	-0.3005	0.6799	0.619
0.5	target	-0.00557	-0.07087	0.01513	0.1951
0.3	target	-2.977e-06	-0.01263	1.443e-05	0.0724
0.0	target	4.6e-07	3.118e-07	3.152e-06	3.932e-06
0.8	best	-0.03767	-0.1241	0.3506	0.4932
0.5	best	-3.005e-05	-0.03306	0.0002353	0.1174
0.3	best	9.793e-07	-0.003058	3.518e-06	0.03883
0.0	best	1.459e-06	1.583e-06	1.048e-06	1.161e-06

SMD 5

β	Variant	UL Med.	UL Mean	LL Med.	LL Mean
0.8	target	-0.1303	-1.29	1.791	11.18
0.5	target	5.663e-06	-0.1738	0.001259	4.929
0.3	target	8.34e-06	-0.001052	5.225e-05	0.08306
0.0	target	3.254e-05	3.52e-05	1.595e-05	1.993e-05
0.8	best	-0.1989	-1.29	1.473	4.267
0.5	best	4.457e-06	-0.292	0.0002627	1.966
0.3	best	1.323e-05	-0.02034	3.015e-05	0.1522
0.0	best	3.444e-05	3.54e-05	2.031e-05	1.972e-05

SMD 6

β	Variant	UL Med.	UL Mean	LL Med.	LL Mean
0.8	target	-2.161	-4.687	7.027	17.94
0.5	target	-0.007443	-1.548	0.04469	8.684
0.3	target	4.215e-06	-0.5741	9.236e-05	3.094
0.0	target	2.898e-05	2.819e-05	4.767e-05	5.152e-05
0.8	best	-1.098	-3.824	3.395	12.62
0.5	best	-0.0007801	-0.4176	0.006854	1.689
0.3	best	6.286e-06	-0.04317	0.0002171	0.2733
0.0	best	3.171e-05	3.662e-05	2.107e-05	2.456e-05

[2] E. Aiyoshi and K. Shimizu. A solution method for the static constrained stackelberg problem via penalty method. *IEEE Trans. on Automatic Control*, 29(12):1111–1114, dec 1984.

[3] G. Anandalingam and D.J. White. A solution method for the linear static stackelberg problem using penalty functions. *IEEE Trans. on Automatic Control*, 35(10):1170–1173, oct 1990.

[4] Jaqueline S. Angelo, Eduardo Krempser, and Helio J. C. Barbosa. Differential evolution for bilevel programming. In *2013 IEEE Congress on Evolutionary Computation*, pages 470–477, 2013.

[5] Jonathan F. Bard. *Practical Bilevel Optimization*. Kluwer Academic Publisher, 1998.

[6] Jonathan F. Bard and James E. Falk. An explicit solution to the multi-level programming problem. *Computers & Operations Research*, 9(1):77–100, 1982.

[7] JonathanF. Bard. Convex two-level optimization. *Mathematical Programming*, 40:15–27, 1988.

[8] Omar Ben-Ayed. Computational difficulties of bilevel linear programming. *Operations Researches*, 38(3):556–560, 1990.

[9] Herminia I. Calvete and Carmen Galé. Solving linear fractional bilevel programs. *Operations Research Letters*, 32:143–151, 2004.

[10] Wilfred Candler and Robert Townsley. A linear two-level programming problem. *Computers & Operations Research*, 9(1):59–76, 1982.

[11] Kalyanmoy Deb. An efficient constraint handling method for genetic algorithms. *Comput. Methods Appl. Mech. Engrg*, 186:311–338, 2000.

[12] Stephan Dempe. *Foundations of Bilivel Programming*. Kluwer Academic Publisher, 2002.

[13] James E. Falk and Jiming Liu. On bilevel programming, part i: General nonlinear cases. *Mathematical Programming*, 70:47–72, 1995.

[14] Alexander I.J. Forrester and Andy J. Keane. Recent advances in surrogate-based optimization. *Progress in Aerospace Sciences*, 45:50–79, 2009.

[15] J.J. Grefenstette and J.M. Fitzpatrick. Genetic search with approximate fitness evaluations. In *Proc. of the Intl. Conf. on Genetic Algorithms and Their Applications*, pages 112–120. Lawrence Erlbaum, 1985.

[16] P. Hansen, B. Jaumard, and G. Savard. New branch-and-bound rules for

TABLE VI. MEDIAN AND MEAN VALUES OF FUNCTION EVALUATIONS (FE) FOR THE UPPER AND LOWER LEVEL PROBLEMS (SMD)

SMD 1

β	Variant	ULFE Med.	ULFE Mean	LLFE Med.	LLFE Mean	%LLSav.
0.8	target	2940	2954	1751000	1751000	76.967
0.5	target	2940	2920	4066000	4052000	46.514
0.3	target	2880	2882	5336000	5292000	29.808
0.0	target	2850	2868	7602000	7611000	-
0.8	best	1800	1814	474000	461600	74.084
0.5	best	1755	1770	1021000	1006000	44.177
0.3	best	1740	1767	1391000	1371000	23.948
0.0	best	1710	1737	1829000	1854000	-

SMD 2

β	Variant	ULFE Med.	ULFE Mean	LLFE Med.	LLFE Mean	%LLSav.
0.8	target	6030	5925	3509000	3417000	53.455
0.5	target	6030	5136	7342000	6521000	2.613
0.3	target	3165	4317	6154000	7424000	18.371
0.0	target	2850	2839	7539000	7518000	-
0.8	best	6031	6031	1271000	1249000	23.664
0.5	best	6031	4968	2637000	2323000	-58.378
0.3	best	2070	3399	1484000	2206000	10.871
0.0	best	1770	1790	1665000	1703000	-

SMD 3

β	Variant	ULFE Med.	ULFE Mean	LLFE Med.	LLFE Mean	%LLSav.
0.8	target	3075	3094	2232000	2245000	74.286
0.5	target	2925	2922	4570000	4538000	47.350
0.3	target	2895	2918	6264000	6245000	27.834
0.0	target	2910	2872	8680000	8653000	-
0.8	best	1860	1874	448800	477700	73.459
0.5	best	1800	1797	1005000	984600	40.568
0.3	best	1770	1759	1254000	1247000	25.843
0.0	best	1710	1736	1691000	1706000	-

SMD 4

β	Variant	ULFE Med.	ULFE Mean	LLFE Med.	LLFE Mean	%LLSav.
0.8	target	6030	5946	3220000	3317000	61.616
0.5	target	6030	5597	6913000	6772000	17.594
0.3	target	3345	4132	6414000	7224000	23.543
0.0	target	3180	3170	8389000	8340000	-
0.8	best	6032	6032	1054000	1028000	29.214
0.5	best	6030	5130	1923000	1815000	-29.147
0.3	best	2250	3346	1322000	1699000	11.216
0.0	best	1951	1963	1489000	1503000	-

SMD 5

β	Variant	ULFE Med.	ULFE Mean	LLFE Med.	LLFE Mean	%LLSav.
0.8	target	6030	6030	7772000	7655000	55.410
0.5	target	6030	5898	17800000	18110000	-2.123
0.3	target	4605	4805	19520000	20600000	-11.991
0.0	target	2895	2884	17430000	17370000	-
0.8	best	6031	5803	1286000	1258000	30.033
0.5	best	5986	4780	2417000	2365000	-31.502
0.3	best	2626	3286	2143000	2273000	-16.594
0.0	best	1740	1727	1838000	1859000	-

SMD 6

β	Variant	ULFE Med.	ULFE Mean	LLFE Med.	LLFE Mean	%LLSav.
0.8	target	6030	6030	2911000	2931000	55.894
0.5	target	6030	4905	5878000	5427000	10.939
0.3	target	3030	3693	4898000	5771000	25.788
0.0	target	2880	2878	6600000	6601000	-
0.8	best	6031	6031	1222000	1203000	17.655
0.5	best	6031	4893	1955000	1958000	-31.739
0.3	best	1980	3045	1259000	1721000	15.162
0.0	best	1680	1692	1484000	1508000	-

linear bilevel programming. *SIAM Journal on Scientific and Statistical Computing*, 13(5):1194–1217, 1992.

[17] Eduardo Krempser, Heder S. Bernardino, Helio J. C. Barbosa, and Afonso C. C. Lemonge. Differential evolution assisted by surrogate models for structural optimization problems. In B. H. V. Topping, editor, *Proc. of the Eighth Intl. Conf. on Engineering Computational Technology*, pages 1–19. Civil-Comp Press, 2012.

[18] V. Oduguwa and R. Roy. Bi-level optimisation using genetic algorithm. In *Proc. of the 2002 IEEE Intl. Conf. on Artificial Intelligence Systems (ICAIS'02)*, ICAIS'02, pages 322–327, Washington, DC, USA, 2002. IEEE Computer Society.

[19] U. Pahner and K. Hameyer. Adaptive coupling of differential evolution and multiquadrics approximation for the tunning of the optimization process. *IEEE Trans. on Magnetics*, 36(4):347–367, 2000.

[20] K. V. Price. An introduction to differential evolution. *New Ideas in Optimization*, pages 79–108, 1999.

[21] J. Rajesh, Kapil Gupta, HariShankar Kusumakar, V.K. Jayaraman, and B.D. Kulkarni. A tabu search based approach for solving a class of bilevel programming problems in chemical engineering. *Journal of Heuristics*, 9:307–319, 2003.

[22] Gilles Savard and Jacques Gauvin. The steepest descent direction for the nonlinear bilevel programming problem. *Operations Research Letters*, 15(5):265–272, 1994.

[23] Donald Shepard. A two-dimensional interpolation function for irregularly-spaced data. In *Proc. of the 1968 23rd ACM National Conference*, pages 517–524, New York, NY, USA, 1968. ACM Press.

[24] K. Shimizu and Min Lu. A global optimization method for the stackelberg problem with convex functions via problem transformation and concave programming. *IEEE Trans. on Systems, Man and Cybernetics*, 25(12):1635–1640, dec 1995.

[25] Kiyotaka Shimizu and Eitaro Aiyoshi. A new computational method for stackelberg and min-max problems by use of a penalty method. *IEEE Trans. on Automatic Control*, AC-26(2):460–466, 1981.

[26] A. Sinha, P. Malo, and K. Deb. Unconstrained scalable test problems for single-objective bilevel optimization. In *Evolutionary Computation (CEC), 2012 IEEE Congress on*, pages 1–8, June.

[27] Y. Wang, Y. Shi, B. Yue, and H. Teng. An efficient differential evolution algorithm with approximate fitness functions using neural networks. In *Proc. of the 2010 Intl. Conf. on Artificial Intelligence and Computational Intelligence: Part II*, AICI'10, pages 334–341, Berlin, Heidelberg, 2010. Springer-Verlag.