This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TEVC.2017.2670659, IEEE Transactions on Evolutionary Computation

A Surrogate Assisted Approach for Single-objective Bilevel Optimization

Md Monjurul Islam, Hemant Kumar Singh and Tapabrata Ray
School of Engineering and Information Technology, University of New South Wales,
Canberra ACT, Australia. Email: md.islam5@student.adfa.edu.au, {h.singh, t.ray}@adfa.edu.au

***Abstract*—Bilevel optimization refers to a hierarchical problem in which optimization needs to be performed at two nested levels, namely the upper level and the lower level. The aim is to identify the optimum of the upper level problem, subject to optimality of the corresponding lower level problem. Several problems from the domain of engineering, logistics, economics and transportation have inherent nested structure which requires them to be modeled as bilevel optimization problems. Bilevel optimization usually requires inordinate amount of function evaluations since a lower level search needs to be conducted for evaluating each upper level solution. The evaluations are especially high when the problems are not suited for exact techniques and evolutionary techniques are employed instead. Reducing this computational effort has been one of the key pursuits in this domain recently. However, the use of surrogate modeling to achieve this goal has so far been scarcely studied. In this paper, we present a surrogate assisted optimization approach towards addressing this research gap. The approach uses surrogates of multiple types in order to provide flexibility of approximating different types of functions more accurately. The algorithm is further strengthened through the use of selective re-evaluation of promising solutions and periodic nested local search. The performance of the proposed algorithm is presented on twenty five standard benchmark problems. The results are compared with a number of other established evolutionary and hybrid algorithms to demonstrate the efficacy of the proposed approach in obtaining competitive results using relatively fewer function evaluations.**

I. Introduction and Background

Bilevel optimization involves solving an upper level problem subject to the optimality of a corresponding lower level problem. The upper and lower level problems are also referred to as the leader and follower problems, respectively. Both levels have their associated objective(s), variable(s) and constraint(s). Such problems model real-life scenarios of cases where the performance of an upper level authority is realizable/sustainable only if the corresponding lower level objective is optimum. A number of practical applications in the field of engineering [1], logistics [2], economics [3] and transportation [4] have inherent nested structure that are suited to this type of modeling.

Mathematically, a single-objective bilevel optimization problem with one leader and one follower objective is defined as shown in Equation 1. The subscript u is used to denote attributes of the upper level problem, whereas subscript l is used for the lower level problem. The upper objective (real-valued) function is $F_u(\mathbf{x_u}, \mathbf{x_l})$ and the lower level objective function is $f_l(\mathbf{x_u}, \mathbf{x_l})$. The vectors $\mathbf{x_u}$ and $\mathbf{x_l}$ denote the upper level variables (in domain $\mathbb{X}_u$) and the lower level variables (in domain $\mathbb{X}_l$) respectively. G and H are the sets of q_u inequality and r_u equality constraints for upper level problem. Similarly, there are q_l inequality constraints in g and r_l equality constraints in h for lower level problem. For a given upper level vector $\mathbf{x_u}$, the evaluation of upper level objective requires the $\mathbf{x_l}$ for the corresponding lower level problem (with $\mathbf{x_u}$ held constant) to be at the optimum. The upper level objective function is optimized with respect to $\mathbf{x_u}$. Lower level objective function is optimized with respect to $\mathbf{x_l}$ using a fixed $\mathbf{x_u}$.

$$
\begin{aligned}
&\underset{\mathbf{x_u}}{\text{Minimize}} \quad F_u(\mathbf{x_u}, \mathbf{x_l}),\\
&\text{S.t.} \qquad G_k(\mathbf{x_u}, \mathbf{x_l}) \leq 0, k = 1, \ldots, q_u,\\
&\qquad\quad H_k(\mathbf{x_u}, \mathbf{x_l}) = 0, k = 1, \ldots, r_u,\\
&\underset{\mathbf{x_l}}{\text{Minimize}} \quad f_l(\mathbf{x_u}, \mathbf{x_l}),\\
&\text{S.t.} \qquad g_k(\mathbf{x_u}, \mathbf{x_l}) \leq 0, k = 1, \ldots, q_l,\\
&\qquad\quad h_k(\mathbf{x_u}, \mathbf{x_l}) = 0, k = 1, \ldots, r_l,\\
&\text{where} \quad \mathbf{x_u} \in \mathbb{X}_u, \quad \mathbf{x_l} \in \mathbb{X}_l
\end{aligned}
\tag{1}
$$

The nested nature of bilevel optimization problems poses a number of additional challenges compared to conventional single-level optimization problems. In particular:

1) The upper level behavior of a bilevel problem may be non-linear, even if the problems at both levels are linear. It is shown in [5] that bilevel linear programming problems are NP hard.
2) Theoretically, an upper level solution is considered valid/feasible only if the corresponding lower level variables are the true global optimum of the lower level problem. Global optimality can be reliably asserted in very limited cases, for example convex and linear problems. However, for most non-linear and black-box problems it is not possible to ensure global optimality.
3) In deceptive cases, an inaccurate lower level optimum may result in an objective value better than true optimum at the upper level, which poses a severe challenge for ranking strategies used within evolutionary optimization techniques.
4) Since a lower level optimization needs to be performed for evaluating each solution at upper level, the number of function evaluations required to solve such problems is usually exorbitantly high. This is particularly the case if the problem is mathematically not suited to be solved using exact techniques, and evolutionary/hybrid techniques are used instead. The issue is further compounded if

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TEVC.2017.2670659, IEEE Transactions on Evolutionary Computation

evaluation of each objective and/or constraint functions is computationally expensive (even mildly).

The subject of bilevel (or more generally, multilevel) programming is not new, and there are references dating back to at least early 1970s [6], [7] dealing with this problem, inspired by the Stackelberg's theory [8] originally proposed in 1952. However, up until about a decade ago, the efforts in this direction primarily considered solving linear or at most quadratic problems at both levels. Furthermore, most of these were solved using exact methods. A comprehensive bibliographic review of prominent works in this area until 1994 was compiled in [9]. More recent works have been reviewed in [10], [11], [12] and tend to use evolutionary methods combined with other heuristic/exact strategies. The use of bilevel optimization has also been recently extended to deal with robust optimization [13], [14].

Some of the salient approaches in the context of single-objective bilevel optimization problems are briefly discussed below.

A. Classical approaches

Some of the classical approaches developed for bilevel problems include simplex method [15], [16], branch and bound [17], descent methods [18] and penalty based approach [19], [20]. The classical methods typically operate by reformulating the problem into a single level problem by incorporating the optimality conditions of the lower level in the formulation. The incorporation could be via an unconstrained formulation (e.g.[20]), or through a complementary constraint using Karush-Kuhn-Tucker (KKT) or other optimality conditions (e.g. [17]). The methods are usually computationally efficient, but often require assumptions on the mathematical properties (such as linearity and convexity) of the functions to be applicable or effective.

B. Evolutionary approaches

The use of metaheuristic methods in bilevel programming is relatively recent. Such methods usually operate using a nested optimization strategy, where a lower level optimization is performed for each upper level variable to assess its performance. Among the earliest effort of this kind, a genetic algorithm based method called GABBA was introduced in [21]. The approach utilized a mutation strategy specific to bilevel problems, and demonstrated improvements in accuracy compared to grid search [22]. A tabu search based algorithm was proposed in [23], and its performance was demonstrated on a set of nine linear/quadratic test problems. Another approach for linear problems was suggested in [24], wherein a genetic algorithm was used to solve a transformed problem using KKT conditions. In [25], a Bilevel Genetic Algorithm (BiGA) was proposed which had a co-evolutionary operator to preserve the interactive nature of the decision making process at two levels. In [26], a genetic algorithm with self-adaptive penalty function was used to solve constrained bilevel optimization problems. The use of Particle Swarm Optimization (PSO) has also been reported to solve bilevel linear programming problems[27], [28]. The performance of most of the above algorithms were demonstrated on bilevel problems with linear or quadratic functions, since the benchmarks had originated from classical literature.

More recent methods have focused on non-linear bilevel optimization problems, which are less tractable due to relatively complex landscapes. Some of the notable contributions in this area are [29], [12], [30], which focused on constructing relatively more complex benchmarks and also proposing evolutionary algorithms to solve them. The set of bilevel test problems (SMD series) proposed in [29] is scalable and first of its kind to incorporate different types of challenges such as non-linearity, multi-modality, and non-cooperation between upper and lower levels. Results of basic Nested Bilevel Evolutionary Algorithm (NBLEA) were also presented. A nested Bilevel Differential Evolution (BIDE) approach was proposed for solving single-objective problems in [31]. Extensive lower level search was performed for each upper level solution, resulting in competitive results but at an expense of high numbers of function evaluations.

C. Hybrid approaches

The idea of using global search methods with classical local search methods has also been explored in a few recent works. Among these, the method presented in [32] used Differential Evolutionary (DE) to solve the leader problem, while interior point method was used for solving the follower problem. Similarly, in [33] DE was used to solve the upper level problem and a gradient search was used for the lower level optimization. Although these schemes led to reduction in the number of function evaluations, there is a considerable likelihood of obtaining a local optimum at the lower level, which in turn could affect the upper level search. To improve on this aspect, a Bilevel Memetic Algorithm (BLMA) that uses global and local searches at both levels during different phases of the search was introduced in [34], [35]. Improvements in performance were demonstrated over other nested strategies.

D. Approximation based approaches with EA

For conventional (single level) optimization problems, it is a common practice to use surrogate/approximate models to guide the search in order to solve a problem with lower number of true function evaluations [36], [37], [38], [39], [40]. This methodology is particularly preferred if each true evaluation is computationally expensive. A review of the methods used for fitness approximation appears in [36], [37] and a survey on the use of fitness approximation in the context of evolutionary computation has been reported in [41]. However, surrogate models have been rarely employed in the domain of bilevel optimization so far. To the authors' knowledge, the only published works that use approximate models during the evolutionary bilevel search are the following:

1) Modified NBLEA: Nested Bilevel Evolutionary Algorithm (NBLEA) was proposed in [29] as a basic nested algorithm in which each of the levels is optimized using an Evolutionary Algorithm (EA). The use of EA at both levels evidently consumes a large number of function evaluations. Therefore the algorithm was subsequently modified and the

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TEVC.2017.2670659, IEEE Transactions on Evolutionary Computation

changes are reflected in the user documentation for its updated code [42]. In the modified algorithm, the lower level problem is first attempted to be modeled as a quadratic programming problem, i.e., quadratic objective function with linear constraints. If the model is built successfully, then a quadratic programming approach is used to solve the (approximate) problem and the solution is accepted as optimal if the true evaluation at the point is close to the approximate optimum. As a result, the function evaluations are significantly reduced if the lower level problems are quadratic in nature (e.g. [31]). However, for more generic non-linear problems which may not be accurately modeled as quadratic programming problems, such benefits may not be realized.

2) BLEAQ: An efficient bilevel evolutionary algorithm based on quadratic approximations (BLEAQ) was proposed in [12]. In this approach, the optimum lower level variable values were approximated as a function of the upper level variables. The model used for fitting was again quadratic. If the quadratic fit was successful, then the optimum lower level variable values were approximated (using the quadratic model) instead of evaluated (through lower level optimization), saving on significant number of lower level evaluations. An extension of the method was proposed in [30], where the search was enhanced using archiving and local search at upper level. More recently, iterative quadratic approximation of lower level optimal value function has been incorporated in [43].

3) Surrogate assisted BIDE: The Bilevel Differential Evolution (BIDE) approach presented in [31] used an extensive DE search at both levels, resulting in a good performance but extremely high number of evaluations. In order to reduce this computational effort, a surrogate model was built between the upper level variables and the corresponding lower level optimum values in [44]. The concept is along the same lines as used in BLEAQ, but the type of surrogate model used in this case was the Similarity-Based Surrogate Model (SBSM) based on k nearest neighbors. The choice of using a true evaluation (lower level optimization) or a surrogate model to identify the lower level optimum was determined through a probability β. As the probability of using the surrogate model (β) was increased, it was observed that the performance of the algorithm deteriorated. For $\beta \geq 0.5$ the results were observed to deviate significantly from the true optimum values.

Thus it can be seen that although preliminary studies have been conducted on demonstrating the benefits of using approximations for bilevel problems, the use of more elaborate surrogate modeling techniques remains rather unexplored. The models used so far tend to be rather simplistic, and thus there remains a significant potential of improving the performance for bilevel problems using more advanced surrogate based strategies.

The above discussion forms the motivation for the study presented in this paper. While the surrogate modeling forms the key idea to reduce the number of function evaluations, it is imperative that inaccuracies in the modeling do not mislead the search at upper/lower levels. Therefore, a number of supporting strategies are included in the proposed approach to overcome the challenges discussed earlier in the section (on Page 1). More specifically:

- We introduce multiple surrogate models to approximate the objective and constraint function(s) at the lower level. These include Response Surface Models (RSM) of order 1 and order 2, and Kriging. This allows for the flexibility of modeling each of these functions separately (using the most appropriate model), as well as possibility of modeling more nonlinear functions. The advantages of using multiple surrogate models for approximation have been highlighted in the context of single level optimization problems in [45], [38] among others.
- Moreover, the modeling and search on the lower level problem are progressive, i.e., the model is updated every few generations during the search through new selectively evaluated solutions.
- In order to reduce the possibility of upper level search being misled, a *re-evaluation* of the top solution in the upper level population is done every generation. Re-evaluation involves an extensive lower level search for a given upper level solution. Since this is done only for selected solutions, the number of evaluations does not grow as high as a conventional nested EA.
- Lastly, a nested local search is performed every few generations in order to expedite the search at the upper level.

The details of the presented approach are discussed in Section II. Numerical experiments are conducted on a set of 25 linear and nonlinear problems with varying complexities, presented in Section III. A comparison is presented with existing strategies (BLMA, NBLEA, BLEAQ and BIDE) in order to highlight the benefits. Concluding remarks are given in Section IV.

II. Proposed approach

In this section, the details of the proposed approach are discussed. We begin with a brief high-level overview of the entire algorithm first, followed by a more detailed description of its key components.

A. Overall framework

An outline of the proposed approach, referred to as Surrogate Assisted Bilevel Algorithm (SABLA) is given in Algorithm 1. At upper level, the proposed framework uses a differential evolution (DE) algorithm. To evaluate each upper level solution, the optimum to the corresponding lower level problem is required, which is obtained through surrogate assisted optimization (SAO). The SAO also uses DE as the underlying algorithm and starts with true evaluation of the initial lower level population. A given set of surrogate models are then built using this data, and the individual models that approximate each objective/constraint the best (based on testing error) are chosen for prediction in subsequent generations of lower level optimization. The population evolved after a given number of generations is truly evaluated again and the surrogate models are updated using the updated archive (i.e., a set of unique solutions that have been truly evaluated so far at the lower level). The intent of this periodic re-training is to ensure that the accuracy of the models as well as choice of

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TEVC.2017.2670659, IEEE Transactions on Evolutionary Computation

the models for any given function improves over the course of search. The process repeats until the prescribed number of lower level generations are exhausted and the best lower level solution obtained is sent to the upper level for evaluation of upper level objective and constraint(s). Towards the end of each generation at upper level, the top upper level solution is "re-evaluated", i.e., an extensive search is conducted at lower level using a hybrid search (global search using differential evolution algorithm followed by local search using interior point algorithm). The intent of re-evaluation is to prevent the algorithm from being misled at upper level on account of inaccurate lower level optima. This is further enforced explicitly in the ranking procedure, where the solutions that have been re-evaluated are ranked above those which haven't in the final ranking of each generation. To expedite the search further, periodically a nested local search is performed starting from a promising upper level solution. The nested local search involves running an IP at both upper and (nested)lower levels. The components of the algorithm are discussed in more detail in the following sub-sections. For the discussions that follow, it is to be noted that (a) at upper level, an "evaluated" solution is one for which the lower level optimum has been obtained through SAO; (b) a re-evaluated solution at upper level is one for which the lower level optimization has been through a more extensive search involving DE followed by IP algorithm; and (c) at lower level, a "truly" evaluated solution is one which is evaluated through lower level objective formulation (and not predicted through a surrogate).

Algorithm 1 Surrogate Assisted Bilevel Algorithm (SABLA)

Require: N_u: Population size for DE at upper level (UL)
T_u: Number of upper level DE generations
N_{nls}: Frequency (no. of UL generations) after which nested local search is invoked

1: Set $i = 1$ {Generation counter}
2: Set $Archive_R = \{\}$ {Archive to store all re-evaluated solutions}
3: Initialize pop_u^i
4: Evaluate pop_u^i (using SAO for lower level)
5: Rank$_{\text{DE}}$ pop_u^i
6: Re-evaluate top solution in pop_u^1, update ranks
7: **for** $i = 2$ to T_u **do**
8: $cpop_u^i$ = Evolve(pop_u^{i-1})
9: Evaluate $cpop_u^i$ (use SAO for lower level)
10: $rpop_u^i$ = $pop_u^{i-1} \cup cpop_u^i$ {Combined parent+child pop}
11: $rpop_u^i$ = Rank$_{\text{DE}}$ ($rpop_u^i$) {Rank parent + child pop}
12: pop_u^i = Reduce ($rpop_u^i$)
13: **if** ($i \mod N_{nls}$) $= 0$ **then**
14: Call nested local search from the top UL solution
15: Re-evaluate the solution obtained.
16: **else**
17: Re-evaluate the existing top ranked UL solution {If top solution has already been evaluated in the past, pick the next ranked solution (and so on)}
18: **end if**
19: Update $Archive_R$
20: pop_u^i = Rank$_{\text{AF}}$ (pop_u^i,$Archive_R$)
21: pop_u^i = Reduce (pop_u^i) {Retain the top N_u solutions}
22: Update best $\mathbf{x_u^*}$, $\mathbf{x_l^*}$, F_u^*, f_l^* as the top-ranked solution in $Archive_R$ based on UL.
23: **end for**
24: Return best F_u^*, f_l^*, $\mathbf{x_u^*}$, $\mathbf{x_l^*}$ in $Archive_R$ based on UL.

B. Global (evolutionary) search

A differential evolution (DE) algorithm has been used as a global search method in the proposed approach. With reference to Algorithm 1, it is used (a) to evolve the upper level variables (Line 8), (b) evolve the population during the lower level SAO (Line 9) and (c) evolve the population during first phase of re-evaluation (Lines 6,15,17).

In this study, the ε-constrained differential evolution (εDE) algorithm, proposed in [46] is used as the underlying DE. The population is initialized using a uniform random distribution in the search space, by creating each design vector as:

$$\mathbf{x_i} = \mathbf{x_{min}} + \mathbf{rand}([\mathbf{0},\mathbf{1}]).(\mathbf{x_{max}} - \mathbf{x_{min}}), \quad i = 1,2,\ldots N, \quad (2)$$

where N is the population size and $\mathbf{x_{max}}$ and $\mathbf{x_{min}}$ are the upper and lower bounds of $\mathbf{x}$.

For evolution, mutation and crossover operators are used. For each population member $\mathbf{x_i}$, mutation is used to generate a trial vector ($\mathbf{x_t}$) using three randomly chosen unique vectors $\mathbf{x_{r1}}, \mathbf{x_{r2}}, \mathbf{x_{r3}}$ from the population in the following way:

$$\mathbf{x_t} = \mathbf{x_{r1}} + F.(\mathbf{x_{r2}} - \mathbf{x_{r3}}), \quad (3)$$

where F is a parameter known as the scaling factor. This trial vector then undergoes crossover with the parent vector $\mathbf{x_i}$. In this algorithm, a binomial crossover with DE/1/rand/bin strategy is used for exploration, which is known to consistently perform well [47]. A child solution is created using binomial crossover of parent vector $\mathbf{x_i}$ and trial vector $\mathbf{x_t}$, by using the following operation for each variable:

$$j_{rand} = \text{randint}(1, N_{var})$$
$$x_c(j) = \begin{cases} x_t(j), & \text{if} \quad rand[0,1) \le CR \quad or \quad j = j_{rand}, \\ x_i(j), & \text{otherwise} \end{cases} \quad (4)$$

Here CR is a user defined parameter, known as crossover probability and N_{var} is the number of variables. The child solution is accepted if it is better than the parent solution. Constraint handling is done based on ε level comparisons, as opposed to explicit *feasibility-first* schemes. For more details, readers are referred to [46]. The global search using the (εDE) is run for a prescribed number of generations (T_u).

C. Local search

In this study, Interior Point Algorithm (IP) based on [48] is chosen to complement the global search. With reference to Algorithm 1, it is used for (a) re-evaluation of a solution by starting from the best solution obtained through DE (Lines 6,15,17), and (b) the nested local search, i.e., a local search at upper level wherein the lower level optimizations are also done using a local search (Line 14).

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TEVC.2017.2670659, IEEE Transactions on Evolutionary Computation

In the IP algorithm presented here, a barrier function is formulated and each of the sub-problems are solved using two powerful techniques: Sequential Quadratic Programming (SQP) and Trust Region (TR) methods. SQP is utilized since it can efficiently handle nonlinear constraints; whereas trust-region strategy allows for handling non-convexity in the problem, in effect "globalizing" the SQP iteration. The strategies used in the algorithm are chosen to satisfy explicit conditions for global convergence stated in [49]. The IP algorithm is used here for doing quick local improvements based on its ability to handle constraints and non-convexity unlike other available local search methods. The implementation of this IP algorithm as available in the `fmincon` package within Matlab 2013b is used in this study. Default parameters for the algorithm (as set in Matlab) are used during optimization.

D. Surrogate Assisted Optimization (lower level)

As discussed in the previous section, the proposed approach attempts to reduce the function evaluations through the use of surrogate models at the lower level.

One of the key considerations in surrogate based optimization is the choice of the model itself. It is reasonably well established in the literature that there isn't a single type of model that can approximate all types of data [38]. Furthermore, a surrogate model that is the best for an objective may not be the best for a constraint of the given problem, or a model that is best for one of the constraints may not be the best for another constraint, and so on. Lastly, if/when more truly evaluated solutions become available during the search, the best model to approximate a particular function may itself change over generations.

Therefore, in the proposed approach, we approximate each objective/constraint function using multiple surrogates. In order to determine the accuracy of a model, the input data (an archive of truly evaluated lower level solutions so far) is split randomly into two parts: training (80% of the data) and testing (20% of the data). The best model for each objective/constraint is chosen as the one with least normalized Mean Squared Error (MSE) on the testing data. In this study, we consider three types of surrogate models: Response Surface Method (RSM) of order 1 (referred to as RSM1), RSM of order 2 (referred to as RSM2), and Kriging. The first two models attempt linear/quadratic fitting of the data, whereas Kriging has been a preferred choice in the literature for generic nonlinear functions. Additionally, we update the models after every few generations, so they capture the evolving landscape more closely.

A brief overview of these models is given below.

1) Response Surface Method (RSM): Response Surface Method or RSM is a linear or polynomial regression. RSM uses first or second degree polynomials to fit the data [50]. A generic second order quadratic polynomial model, with m input variables $\{x_1, x_2, \ldots x_m\}$ can be written as

$$y(\mathbf{x}) = \beta_0 + \sum_{i=1}^{m} \beta_i x_i + \sum_{i=1}^{m} \beta_{ii} x_i^2 + \sum_{i=1}^{m-1} \sum_{j=i+1}^{m} \beta_{ij} x_i x_j \qquad (5)$$

where $\{\beta_0, \beta_i, \beta_{ii}, \beta_{ij}\}$ are the unknown parameters of the model that are determined from the given data. In vector form, this can be written as $y(\mathbf{x}) = \mathbf{f}^T\mathbf{b}$. The vector $\mathbf{f}$ contains all the terms of $x_1, x_2, \ldots, x_m$ and vector $\mathbf{b}$ contains all the unknown coefficients. The value of unknown coefficients is determined using least squares method. The least squares estimate of $\mathbf{b}$ is given by,

$$\hat{\mathbf{b}} = (\mathbf{F}^T\mathbf{F})^{-1}\mathbf{F}^T\mathbf{Y}, \qquad (6)$$

where $\mathbf{F}$ is a matrix containing N rows, each row is a vector $\mathbf{f}^T$ evaluated at an observation and $\mathbf{Y}$ are the observed responses. In this study, we consider RSM of orders 1 and 2.

2) Kriging: Kriging, also known as Design and Analysis of Computer Experiments (DACE) [51], [52] is among the most popular methods to approximate generic non-linear functions. It attempts to approximate the function as a combination of a global regression model and a deviation term (with zero mean):

$$y(\mathbf{x}) = \mu(\mathbf{x}) + \varepsilon(\mathbf{x}) \qquad (7)$$

The regression model is typically polynomial, for example the one shown in Equation 5. The covariance function is given using the equation:

$$cov(\varepsilon(\mathbf{x_i}), \varepsilon(\mathbf{x_j})) = \sigma.^2 R(\mathbf{x_i}, \mathbf{x_j}) \qquad (8)$$

Here, σ^2 is the process variance and $R(\mathbf{x_i}, \mathbf{x_j})$ is a spatial correlation function, typically modeled as

$$R(\mathbf{x_i}, \mathbf{x_j}) = exp(-\sum_{k=1}^{m} \theta_k |x_k^i - x_k^j|^p) \qquad (9)$$

Here, θ_k and p are hyper-parameters that are determined using the maximum likelihood estimation (MLE) in order to obtain the Kriging model. For more detailed description of the model, the readers are referred to [51].

The overall SAO is outlined in Algorithm 2 and proceeds as follows. For a given $\mathbf{x_u}$ (which remains fixed), a set of lower level (LL) solutions are generated. The size of the set is equal to the lower level DE population size (N_l). The lower level objective and constraint function(s) are truly evaluated using these solutions, and added to an archive ($Archive_{ll}$) of the unique truly evaluated solutions. Thereafter, each of the surrogate models (RSM1, RSM2, Kriging) are built using the archive and best model obtained for each objective/constraint function are chosen for predictions during subsequent generations. The solutions are then evolved using the approximated functions for a prescribed number of generations (referred to as surrogate rebuild frequency $Surr_{RF}$). The population obtained is then truly evaluated, and added to $Archive_{ll}$. The models are re-built again using the updated $Archive_{ll}$. The DE then proceeds on the updated model for the same prescribed number of generations, and the process repeats until the given lower level number of generations (T_l) are exhausted. The best solution obtained is then truly evaluated and returned to the upper level for evaluation of upper level objective and constraint(s).

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TEVC.2017.2670659, IEEE Transactions on Evolutionary Computation

Algorithm 2 Surrogate assisted optimization (SAO) for lower level problem

Require: N_l: Population size for lower level Surrogate Approximation
T_l: Maximum number of lower level generations
$\mathbf{x_u}$: Upper level variables (fixed parameters for lower level problem)
$Surr_{RF}$: Surrogate rebuild frequency (generations)
$Surr_{set}$: Set of surrogates for approximation {RSM1, RSM2, Kriging}

1: Set $i = 1$ {Generation counter for DE}
2: Set $Archive_{ll} = \{\}$ {Archive to store all true evaluated LL solutions}
3: Initialize (pop_l^i)
4: Evaluate (pop_l^i) {True evaluations of LL solutions}
5: Update $Archive_{ll}$
6: $Surr \leftarrow$ Build and choose best surrogates for each LL objective/constraint(s) using $Archive_{ll}$
7: **for** $i = 2$ to T_l **do**
8: $cpop_l^i$ = Evolve (pop_l^{i-1})
9: Evaluate$_{\mathrm{A}}$ ($cpop_{l,i}$,$\mathbf{x_u}$,$Surr$) {Predict LL objective/constraint(s) using best surrogate(s)}
10: $rpop_l^i = cpop_l^i \cup pop_l^{i-1}$ {Combined parent + child pop}
11: $rpop_l^i$ = Rank$_{\mathrm{DE}}$ ($rpop_l^i$)
12: pop_l^i = Reduce ($rpop_l^i$) {Select top N_l best solutions}
13: Update best f_l and corresponding $\mathbf{x_l}$
14: **if** ($i \mod Surr_{RF}) == 0$ **then**
15: Evaluate (pop_l^i) {True evaluations of LL solutions}
16: Update $Archive_{ll}$
17: Update $Surr$ {Re-build surrogates on $Archive_{ll}$}
18: **end if**
19: **end for**
20: Return $\mathbf{x_l^*}$, f_l^* (true evaluation).

E. Re-evaluation

As discussed in the previous section, one of the key challenges in bilevel optimization is that an inaccurate lower level optimum may result in a superior upper level objective value compared to the true lower level optimum. This creates a challenge for ranking at the upper level, since it would be difficult to remove such spurious solutions due to their superior UL objective values. For certain deceptive problems, this property may mislead the upper level search drastically towards a sub-optimal region.

While this possibility cannot be eliminated theoretically (since optimality cannot be guaranteed for most cases), we use a stricter check on the highly ranked solution(s) in the population to reduce the possibility of this happening. This is done through a *re-evaluation* step, wherein the top ranked upper level solution undergoes a more extensive lower level search (Algorithm 1, Lines 6,15,17). The search operates on true lower level evaluations (and not on surrogate), and comprises a global search using DE first, followed by a further local search using IP from the best solution obtained using DE. This process consumes relatively high number of function evaluations, but since it is done selectively on only the top ranked solutions (instead of on all solutions), the total number of overall evaluations are still significantly low compared to conventional nested evolutionary approaches. Its advantage in preventing the search being misguided outweighs the cost of these evaluations, as demonstrated later in numerical experiments (Section III).

There may be instances where the top ranked solution in a population has already been evaluated previously. All re-evaluated solutions are tracked through the use of an archive ($Archive_R$). If the top solution in a population identified to be re-evaluated already exists in the archive, then the next best solution in the population is considered for re-evaluation instead (and so on).

F. Nested local search

To expedite the upper level search, a nested local search is performed after a every few generations. During this process, IP is used to improve the objective value at the upper level, while also using the IP for lower level optimization. The top individual from the upper level population is chosen as the starting solution for the upper level search. For each of the lower level searches, the initial lower level solution is identified as follows. From the archive of re-evaluated solutions, the solution closest to the current $\mathbf{x_u}$ is identified. The $\mathbf{x_l^*}$ corresponding to this $\mathbf{x_u}$ is then chosen as the starting point. This method for choosing the starting lower level solution has also been earlier adopted in [34]. This relies on the assumption that two solutions which are close at the upper level are likely to have close lower level optima.

The solution obtained through the above nested local search undergoes re-evaluation before being added to the population again. Subsequently, it is also added to the archive of re-evaluated solutions ($Archive_R$).

G. Ranking

The last key component of the proposed approach is the ranking of solutions. While the DE ranking (referred to as Rank$_{\mathrm{DE}}$ in Algorithm 1) is standard and done in the same way as in [46], the upper level ranking is further enhanced by different treatment of re-evaluated solutions.

The re-evaluated solutions represent the most promising solutions encountered so far during the search. As discussed above, an archive of all such re-evaluated ($Archive_R$) solutions is maintained and continually updated whenever a new solution is re-evaluated. The confidence in the upper level objective values of these solutions is evidently high due the extensive lower level search. Therefore, just before the end of the generation (Line 20 in Algorithm 1), the final ranking of the solutions involves placing all the solutions of $Archive_R$ ahead of the other solutions. Among the $Archive_R$ solutions, the solutions are ordered according to feasibility first scheme. The remaining solutions (which have not been re-evaluated) retain the order obtained by DE ranking. This process is referred to as Rank$_{\mathrm{AF}}$ (i.e., Ranking with $Archive_R$ first) in Algorithm 1 (Line 20). Thereafter, the top N_u solutions are selected as members of the parent population for the next generation.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TEVC.2017.2670659, IEEE Transactions on Evolutionary Computation

III. Numerical Experiments

A. Comparison metrics

In order to evaluate and compare the performance of the proposed approach, the metrics outlined below are used. Due to the nested nature, the comparison of performance is not as straightforward for bilevel optimization problems. These challenges are also highlighted below.

1) Objective values: For the single level optimization problems, usually the performance can be judged by the statistics of the solutions obtained by each algorithm across multiple independent runs. For minimization, the algorithm which shows lower objective values statistically is considered to have better performance. However, this method of comparison does not hold for bilevel problems as an inaccurate optimum at lower level may result in better objective value for upper level than the true optimum. Thus, in order to unambiguously compare the performance, the availability of true optimum is essential. When the optima are known (which is the case for the benchmark problems), the performance is compared using an error metric, i.e., for upper level $\varepsilon_u = |F_u^* - F_u^{*t}|$, where F_u^* is the best value obtained by the algorithm at the end of the run, and F_u^{*t} is the true optimum. Similarly, for the lower level, the $\varepsilon_l = |f_l^* - f_l^{*t}|$. Lower values of ε_u and ε_l represent better accuracy.

2) Computational effort: The next challenge lies in comparing the computational effort, as the upper and lower level have different number of function evaluations. This instigates a question about which of them should be compared or prescribed to a specific number for fair comparison. Due to the nested nature, and with different strategies of handling the function evaluations, it is almost always unattainable to terminate the algorithm at the same number of both upper and lower function evaluations. Hence in the numerical experiments that follow, the population size has been set the same for fairness, but due to different strategies and termination criteria being used at both levels, the eventual numbers of function evaluations could end up quite different.

It is to be noted that for real life examples, this comparison could also depend on the relative computational complexity involved in evaluating a function at upper and lower levels. If upper level evaluation is computational intensive, then the total time taken will depend heavily on the upper level function evaluations. Same goes for lower level comparisons.

3) Performance profiles: Apart from reporting the median accuracy of the algorithms on individual problems, *performance profiles* [53], [54] are also used for a visual comparison. The performance profile is a statistical tool for comparing the performance of multiple approaches against a given performance metric for a large set of problems. In a performance profile plot, the horizontal axis (τ) signifies the performance values relative to the best performing algorithm for a problem. The vertical axis ($\rho(\tau)$) signifies the cumulative distribution of the performance. For a given value of τ, it indicates what proportion of problems was an algorithm able to solve within a factor τ of the best performer. The algorithms could thus be compared based on a given level of performance τ. Additionally, overall performance could also be quantified using the area under the curve ($AUC = \int \rho(\tau) d\tau$) of the profile, larger the better.

B. Test Problems

The test problems used in this study are a collation of two sets of benchmarks. The first set consists of mostly linear/quadratic bilevel problems from the classical literature. For ease of reference, they are names BLTP (Bilevel Test Problems) series. The second set comprises more recently proposed SMD series problems. SMD is a scalable set of test problems which are designed to cover a wide range of difficulties associated with bilevel optimization, including non-linearity, multi-modality and conflict in upper and lower level objectives. The problems, their source and their optimum values are listed in Table I. For SMD problems, the number of variables at upper and lower levels are set to 2 and 3 for all problems. The mathematical formulations are omitted for the sake of brevity, and could be found in the cited references.

TABLE I
Benchmark problems considered in this study

Problem Name	Source	(F_u^*, f_l^*)
BLTP1	[20] (Problem 2)	(0,200)
BLTP2	[55] (Problem 1)	(17,1)
BLTP3	[31] (Problem 14)	(1,0)
BLTP4	[25] (F1)	(1000,1)
BLTP5	[56]	(-1.4074,7.6172)
BLTP6	[55] (Problem 3)	(100,0)
BLTP7	[57]	(49,-17)
BLTP8	[58]	(-1,0)
BLTP9	[23] (Problem 4)	(85.0909,-50.1818)
BLTP10	[23] (Problem 2)	(5,4)
BLTP11	[23] (Problem 3)	(9,0)
BLTP12	[17] (Problem 2)	(3.25,4)
BLTP13	[59]	(29.2,-3.2)
SMD1	[29]	(0,0)
SMD2	[29]	(0,0)
SMD3	[29]	(0,0)
SDM4	[29]	(0,0)
SDM5	[29]	(0,0)
SMD6	[29]	(0,0)
SMD7	[29]	(0,0)
SMD8	[29]	(0,0)
SMD9	[29]	(0,0)
SMD10	[29]	(4,3)
SMD11	[29]	(-1,1)
SMD12	[29]	(3,4)

C. Experimental settings and algorithms used for comparison

In order to assess the performance of the proposed algorithm, we compare its results with the key existing algorithms in the area. These include BLMA [34], NBLEA [42], BLEAQ [12] and BIDE [44]. For BIDE, two versions are considered: BIDE(T) which operates entirely on true function evaluations, and BIDE(S) which operates mostly ($\beta = 0.8$) on surrogates. The individual experimental settings for each of the algorithms are discussed below:

1) SABLA: For the proposed algorithm, the population size and number of generations are set to 20 and 50 respectively for both levels. A maximum of 250 function evaluations is used for each individual local

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TEVC.2017.2670659, IEEE Transactions on Evolutionary Computation

search (IP). The frequency of conducting the nested local search (N_{nls}) and rebuilding surrogates during SAO at lower level ($Surr_{RF}$) are both set to 10. For DE, the crossover rate CR and scaling factor F are set to 0.9 and 0.7 respectively. For the surrogates, 80% of the data (archive of unique truly evaluated lower level solutions so far for the given $\mathbf{x_u}$) is used for training and 20% for testing. The parameters were selected (after some preliminary experiments) so as to result in roughly similar orders of function evaluations generated by the other algorithms. However, as previously mentioned, with different strategies used by each algorithm (including termination criteria), it is very difficult to ensure same number of evaluations at each level as reflected in the experiments.

2) BLMA [34]: A population size of 20 is used at both levels. The upper level DE generations are set to 20 for SMD9-SMD12 (which are relatively harder problems), and 15 for others. The lower level DE generations are set to 50. The switch from global to local search for lower level optimization is done after the 16 and 13 generations for the two different settings respectively. A maximum of 250 function evaluations are used for each local search.
3) NBLEA [29]: The implementation available for download at http://bilevel.org is used in this study, with population size 20 and default parameter settings. As discussed in Section I, the current version of the code is different from the one used for the study in [29]. It uses quadratic programming at the lower level wherever possible to reduce the number of function evaluations.
4) BLEAQ [12]: The implementation available for download at http://bilevel.org is used in this study, with population size 20 and default parameter settings.
5) BIDE [44]: For BIDE, the results reported in [44] have been directly used for comparison, since the code for the algorithm is not available. The study presented in [44] includes experiments on all BLTP problems and SMD1-SMD6 problems considered in this paper.

For each problem, twenty nine independent runs are performed using each of the algorithms (except BIDE) and statistics are reported in terms of the performance measures discussed above. While reporting the median errors, we consider a precision of 1×10^{-6}. Consequently, if the error is less than 1×10^{-6}, we simply set it to 1×10^{-6}. This is in order to avoid comparison between extremely small values, where the difference in values is not significant. A statistical comparison of the results using other algorithms against SABLA is done using Wilcoxon signed rank test with 95% confidence level, indicated as (=), (+) or (-) for equivalent, higher or lower values (note that the rank sum tests are not presented for BIDE as the raw data for the runs is not available).

D. Results

1) Objective values and function evaluations: The statistics for the results ($\varepsilon_u, \varepsilon_l$) obtained using all five (six considering two versions of BIDE) algorithms are shown in Table II, whereas the median function evaluations used by the algorithms are reported in Table III.

The best median values obtained for upper and lower level are indicated in bold in Table II. It can be seen that SABLA is able to obtain the lowest error values for 18/25 problems at upper level and 20/25 problems at lower level. For BLMA, NBLEA and BLEAQ, these numbers are {13,17}, {7,9} and {8,8} respectively. BIDE(T) obtains the best results for 6/19 problems at upper level and 4/19 at lower level, whereas BIDE(S) obtains the best results for 3/19 at upper and 4/19 at lower level (results for SMD7-SMD12 are not available for BIDE). SABLA is also able to reach the threshold error of 1×10^{-6} more often than all other algorithms. For the remaining problems, it still achieves reasonably competitive results, especially considering the number of function evaluations. For example, for SMD11 it is seen that BLMA, NBLEA and BLEAQ all use much higher number of function evaluations compared to SABLA in order to obtain better results. For BLTP6, SABLA is only outperformed by BIDE(T) with extensive search. For BLTP10, it is only marginally worse than the best performing algorithms (BLMA at lower and NBLEA/BLEAQ at upper). Similarly for BLTP13, the results are marginally inferior to BLMA, whereas the remaining algorithms show significantly worse performance.

A particular case where most of the algorithms could not reach the vicinity of the true optimum was BLTP4. The apparent reason for this behavior is that for the true upper level optimum x_u (=0), there exist infinite possibilities to construct a global optimum for lower level problem. For example $\mathbf{x_l}$ = {1,0},{0.556,0.444}, {0.588,0.412}, etc. all satisfy the constraints $x_{l1}-x_{l2}\leq 1$, $x_{l1}+x_{l2}\leq 1$ and result in global lower level optimum value of $f_l = x_{l1}+x_{l2} = 1$. However, each of these result in a different F_u value at upper level, and only $\{1,0\}$ results in upper level global optimum. This is also evidenced by the fact that f_l is at the true optimum for all algorithms, but F_u deviates significantly from the true optimum. For this case, the extensive search of BIDE was able to obtain the closest solution to the true optimum at upper level, albeit with much higher number of function evaluations compared to others.

The median function evaluations used by each of the algorithms are shown in Table III and Figure 3. As expected, BIDE(T) which performs and extensive DE search at both levels results in the highest number of function evaluations. These are brought down to some extent through the use of surrogates in BIDE(S), but with significant trade-off in performance as seen in Table II. The function evaluations used by NBLEA and BLEAQ show an interesting trend. For BLTP problems, it can be seen that the function evaluations (especially at lower level) are significantly lower compared to those for SMD problems. This is because as discussed in Section I, NBLEA and BLEAQ attempt to solve the lower level problem as a quadratic programming problem first. If this optimization is successful, they accept it as the lower level optimum, whereas if not, then a more elaborate global search method (EA) is used. Since for BLTP problems, the functions at both levels are linear/quadratic, this approach was able to use quadratic approximations for search. However, for SMD

TABLE II
MEDIAN ACCURACY ($\varepsilon_u, \varepsilon_l$) OBTAINED USING NBLEA, BLEAQ, BIDE(T), BIDE(S), BLMA AND SABLA ON 25 STANDARD BILEVEL PROBLEMS

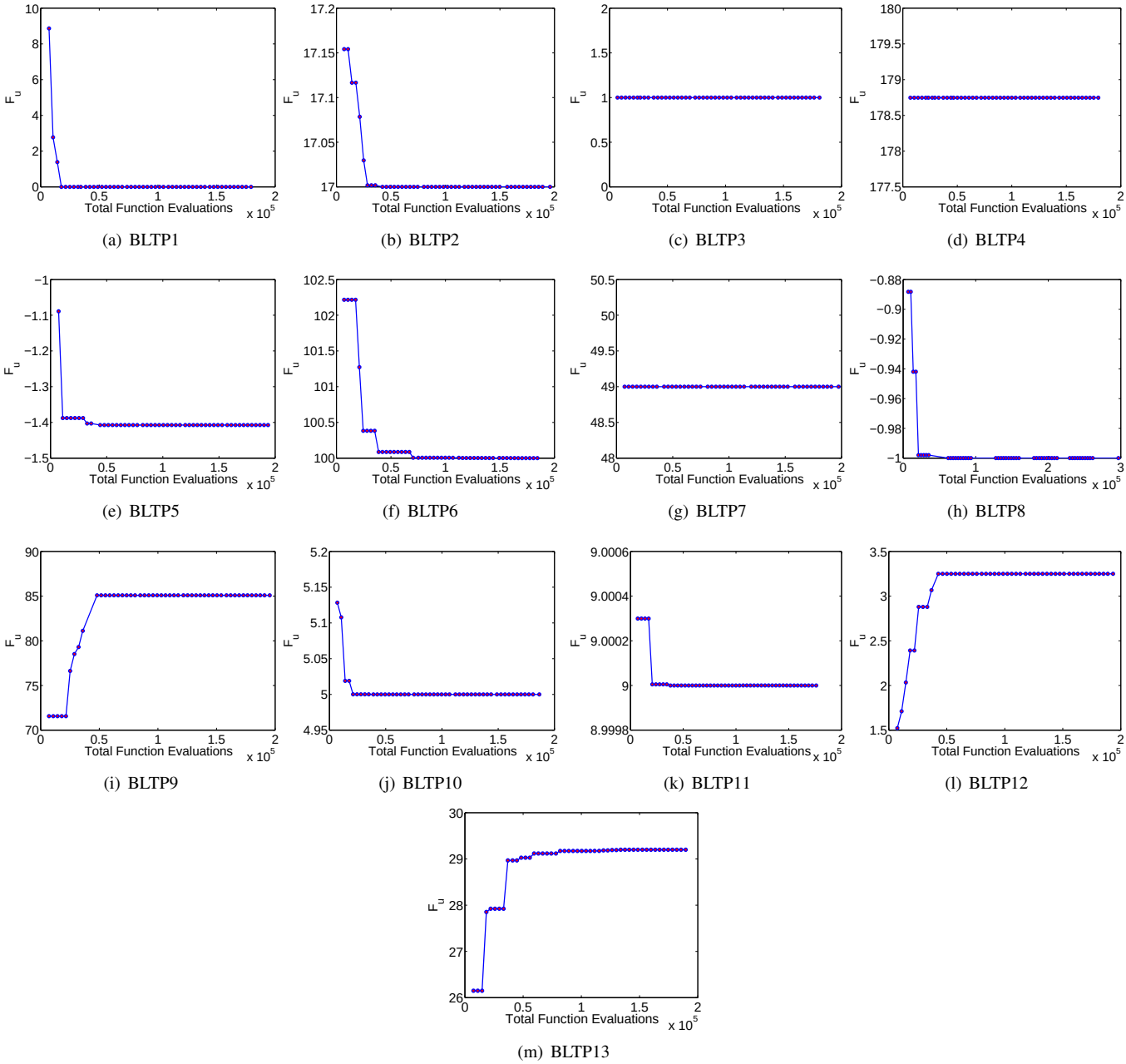
Prb No	SABLA		BLMA		NBLEA		BLEAQ		BIDE(T)		BIDE(S)	
	UL Acc	LL Acc	UL Acc	LL Acc	UL Acc	LL Acc	UL Acc	LL Acc	UL Acc	LL Acc	UL Acc	LL Acc
BLTP1	**1.00E-06**	**1.00E-06**	**1.00E-06**(=)	**1.00E-06**(=)	2.13E-04(+)	1.00E+02(+)	7.96E-03(+)	1.00E+02(+)	**1.00E-06**	**1.00E-06**	**1.00E-06**	**1.00E-06**
BLTP2	1.34E-06	**3.53E-06**	2.00E-06(=)	5.00E-06(+)	2.01E-03(+)	5.25E-03(+)	8.23E-03(+)	2.12E-02(+)	**1.00E-06**	1.10E-03	6.80E-01	3.82E-01
BLTP3	**1.00E-06**	**1.00E-06**	**1.00E-06**(=)	**1.00E-06**(=)	**1.00E-06**(=)	**1.00E-06**(=)	**1.00E-06**(=)	**1.00E-06**(=)	**1.00E-06**	**1.00E-06**	**1.00E-06**	**1.00E-06**
BLTP4	8.21E+02	**1.00E-06**	5.61E+02(=)	**1.00E-06**(=)	3.91E+02(-)	**1.00E-06**(=)	3.78E+02(-)	**1.00E-06**(=)	**1.00E-06**	**1.00E-06**	4.87E+01	**1.00E-06**
BLTP5	**4.46E-06**	**7.77E-05**	7.00E-06(+)	8.40E-05(+)	3.49E-03(+)	7.31E-03(+)	8.43E-03(+)	1.77E-02(+)	4.00E-04	1.20E-03	1.37E-01	3.31E-01
BLTP6	1.75E-04	**1.00E-06**	3.79E-02(+)	**1.00E-06**(=)	9.30E-03(+)	**1.00E-06**(+)	1.75E-02(+)	1.02E-06(+)	**1.00E-06**	2.21E-06	2.50E-01	2.96E-03
BLTP7	**1.00E-06**	**1.00E-06**	6.00E-06(+)	3.00E-06(+)	**1.00E-06**(=)	**1.00E-06**(=)	**1.00E-06**(=)	**1.00E-06**(=)	4.00E-02	2.00E-02	1.60E-01	1.10E-01
BLTP8	**3.42E-05**	**1.00E-06**	1.90E-04(+)	**1.00E-06**(=)	7.62E-05(+)	**1.00E-06**(=)	3.23E-04(+)	**1.00E-06**(=)	1.50E-02	3.04E-04	2.50E-02	1.13E-03
BLTP9	**6.51E-06**	1.72E-05	7.00E-06(=)	**1.70E-05**(=)	6.82E-03(+)	2.37E-03(+)	2.96E-02(+)	1.03E-02(+)	7.09E-02	3.18E-02	3.11E-01	1.12E-01
BLTP10	2.63E-06	4.65E-04	5.00E-06(+)	**8.70E-05**(-)	**1.00E-06**(-)	1.60E-03(+)	**1.00E-06**(-)	1.33E-03(=)	3.00E-03	2.50E-02	2.34E-01	1.55E-01
BLTP11	**1.00E-06**	**1.00E-06**	**1.00E-06**(=)	**1.00E-06**(=)	**1.00E-06**(+)	**1.00E-06**(=)	**1.00E-06**(=)	**1.00E-06**(=)	**1.00E-06**	**1.00E-06**	**1.00E-06**	**1.00E-06**
BLTP12	**1.00E-06**	**1.00E-06**	**1.00E-06**(=)	2.00E-06(+)	x	x	x	x	3.00E-03	5.50E-02	7.70E-02	1.54E+00
BLTP13	6.01E-04	4.16E-05	**2.50E-05**(=)	**1.00E-06**(=)	8.89E-02(+)	1.72E-02(+)	2.07E-01(+)	4.01E-02(+)	2.90E-01	2.60E-02	7.70E-01	4.60E-02
SMD1	**1.00E-06**	**1.00E-06**	**1.00E-06**(=)	**1.00E-06**(=)	2.44E-05(+)	1.02E-05(+)	**1.00E-06**(=)	**1.00E-06**(=)	3.79E-05	1.83E-05	4.85E-05	2.69E-05
SMD2	**1.00E-06**	**1.00E-06**	**1.00E-06**(=)	**1.00E-06**(=)	1.03E-05(+)	3.93E-06(+)	3.07E-06(+)	3.51E-06(+)	1.18E-05	7.20E-06	1.85E+00	4.70E+00
SMD3	**1.00E-06**	**1.00E-06**	**1.00E-06**(=)	**1.00E-06**(=)	1.89E-05(+)	1.10E-05(+)	8.80E-06(+)	6.30E-06(+)	3.71E-05	1.50E-05	3.73E-05	1.86E-05
SMD4	**1.00E-06**	**1.00E-06**	**1.00E-06**(+)	**1.00E-06**(=)	1.43E-05(+)	1.64E-05(+)	**1.00E-06**(+)	3.25E-06(+)	1.46E-06	1.05E-06	3.77E-02	3.77E-02
SMD5	**1.00E-06**	**1.00E-06**	**1.00E-06**(=)	**1.00E-06**(=)	**1.00E-06**(+)	**1.00E-06**(+)	**1.00E-06**(=)	**1.00E-06**(=)	3.44E-05	2.03E-05	1.99E-01	1.47E+00
SMD6	**1.00E-06**	**1.00E-06**	**1.00E-06**(=)	**1.00E-06**(=)	**1.00E-06**(+)	**1.00E-06**(+)	**1.00E-06**(=)	**1.00E-06**(=)	3.17E-05	2.11E-05	1.10E+00	3.40E+00
SMD7	**1.00E-06**	**1.00E-06**	**1.00E-06**(=)	**1.00E-06**(=)	3.23E-05(+)	1.23E-06(+)	5.74E-06(+)	6.00E-06(+)	x	x	x	x
SMD8	2.76E-03	6.77E-04	2.45E-03(-)	9.35E-04(=)	**1.22E-04**(-)	**7.42E-05**(-)	1.29E-03(-)	3.21E-04(-)	x	x	x	x
SMD9	**1.78E-06**	**1.77E-06**	1.00E-05(+)	1.30E-05(+)	2.47E-05(+)	1.45E-05(+)	3.44E-06(+)	4.88E-06(+)	x	x	x	x
SMD10	**3.80E-06**	**3.90E-06**	1.60E+01(+)	1.60E+01(+)	1.71E-02(=)	3.15E-02(+)	3.81E-03(=)	3.64E-03(+)	x	x	x	x
SMD11	2.31E-01	3.25E-01	**2.00E-06**(-)	**2.00E-06**(-)	2.67E-04(-)	5.68E-04(-)	5.38E-04(-)	9.80E-04(-)	x	x	x	x
SMD12	**1.00E-06**	**1.00E-06**	4.00E-06(-)	1.60E+01(+)	2.59E-02(-)	5.16E-02(-)	1.27E-01(=)	4.59E-02(-)	x	x	x	x

TABLE III
MEDIAN FUNCTION EVALUATIONS USING NBLEA, BLEAQ, BIDE(T), BIDE(S),BLMA AND SABLA ON 25 STANDARD BILEVEL PROBLEMS. FOR BLMA, THE NUMBERS IN ITALICS DENOTE THE MEDIAN EVALUATIONS REQUIRED BY SABLA TO REACH THE BEST ACCURACY OF 10^{-6} AT BOTH LEVELS.

Prb No	SABLA		BLMA		NBLEA		BLEAQ		BIDE(T)		BIDE(S)	
	UL FE	LL FE	UL FE	LL FE	UL FE	LL FE	UL FE	LL FE	UL FE	LL FE	UL FE	LL FE
BLTP1	1.30E+03	1.79E+05	3.24E+02	2.60E+05	4.73E+02	8.35E+03	4.98E+02	2.28E+03	6.03E+03	2.74E+06	6.04E+03	6.13E+05
	8.40E+01	*1.40E+04*										
BLTP2	1.27E+03	1.89E+05	4.14E+02	2.66E+05	1.43E+03	1.36E+04	8.89E+02	2.73E+03	6.03E+03	2.54E+06	6.03E+03	4.95E+05
BLTP3	1.62E+03	1.80E+05	3.23E+02	2.60E+05	1.06E+02	9.95E+02	1.23E+02	2.84E+02	4.20E+02	1.65E+05	4.53E+02	7.06E+04
	4.20E+01	*6.86E+03*										
BLTP4	1.17E+03	1.78E+05	3.59E+02	2.70E+05	2.44E+02	4.35E+03	5.33E+02	1.21E+03	7.06E+03	2.32E+07	6.09E+03	2.60E+06
BLTP5	1.28E+03	1.92E+05	3.98E+02	2.65E+05	8.38E+02	1.49E+04	1.67E+02	1.47E+03	5.53E+02	2.67E+05	7.74E+02	1.13E+05
BLTP6	1.26E+03	1.82E+05	3.29E+02	2.61E+05	7.08E+02	6.63E+03	8.30E+02	1.78E+03	6.31E+02	3.80E+05	8.40E+02	1.40E+05
BLTP7	1.57E+03	1.95E+05	3.55E+02	2.62E+05	1.36E+02	1.26E+03	1.29E+02	2.99E+02	6.53E+02	3.21E+05	9.00E+02	1.23E+05
	4.20E+01	*7.18E+03*										
BLTP8	2.11E+03	2.95E+05	4.34E+02	2.77E+05	6.37E+02	1.14E+04	5.64E+02	2.50E+03	7.28E+02	5.86E+05	9.02E+02	1.96E+05
BLTP9	1.29E+03	1.95E+05	4.06E+02	2.71E+05	1.03E+03	9.70E+03	1.72E+02	7.40E+02	7.41E+02	3.65E+05	1.11E+03	1.48E+05
BLTP10	1.69E+03	1.83E+05	4.66E+02	2.62E+05	8.47E+02	7.94E+03	8.99E+02	7.72E+02	6.15E+02	4.82E+05	6.03E+03	8.22E+05
BLTP11	1.31E+03	1.75E+05	3.31E+02	2.60E+05	5.21E+02	4.89E+03	5.00E+02	4.07E+02	5.10E+02	2.21E+05	4.80E+02	6.59E+04
	2.41E+02	*3.77E+04*										
BLTP12	1.25E+03	1.91E+05	3.38E+02	2.70E+05	x	x	x	x	7.58E+02	2.74E+06	1.26E+03	1.07E+06
	8.40E+01	*1.45E+04*										
BLTP13	1.15E+03	1.93E+05	4.41E+02	2.70E+05	5.55E+02	1.50E+04	1.50E+02	1.87E+03	9.63E+02	2.30E+06	1.30E+03	7.69E+05
SMD1	1.18E+03	1.81E+05	3.70E+02	2.73E+05	5.92E+02	2.55E+05	5.83E+02	1.22E+05	1.71E+03	1.83E+06	1.80E+03	4.74E+05
	3.42E+02	*5.47E+04*										
SMD2	1.21E+03	1.94E+05	4.03E+02	2.75E+05	5.65E+02	2.60E+05	5.40E+02	1.73E+05	1.77E+03	1.67E+06	6.03E+03	1.27E+06
	3.54E+02	*5.24E+04*										
SMD3	1.18E+03	1.80E+05	4.32E+02	2.86E+05	5.98E+02	2.98E+05	5.23E+02	1.42E+05	1.71E+03	1.69E+06	1.86E+03	4.49E+05
	3.42E+02	*5.78E+04*										
SMD4	2.32E+03	2.93E+05	5.71E+02	3.07E+05	5.56E+02	2.35E+05	5.18E+02	1.33E+05	1.95E+03	1.49E+06	6.03E+03	1.05E+06
	4.82E+02	*7.99E+04*										
SMD5	2.32E+03	2.38E+05	5.71E+02	2.98E+05	1.27E+03	6.82E+05	8.50E+02	3.72E+05	1.74E+03	1.84E+06	6.03E+03	1.29E+05
	4.82E+02	*7.92E+04*										
SMD6	1.39E+03	2.25E+05	3.95E+02	2.80E+05	1.41E+03	3.83E+04	1.56E+03	7.26E+03	1.68E+03	1.48E+06	6.03E+03	1.22E+06
	4.82E+02	*8.22E+04*										
SMD7	1.21E+03	1.95E+05	4.11E+02	2.77E+05	5.89E+02	2.48E+05	4.73E+02	1.42E+05	x	x	x	x
	3.54E+02	*5.38E+04*										
SMD8	2.33E+03	3.15E+05	5.71E+02	2.99E+05	4.12E+03	2.30E+06	1.29E+03	6.71E+05	x	x	x	x
SMD9	1.61E+03	2.48E+05	5.46E+02	3.63E+05	1.16E+03	6.55E+05	5.44E+02	1.98E+05				
SMD10	1.21E+03	1.92E+05	4.59E+02	3.45E+05	1.84E+03	1.71E+06	7.01E+02	2.58E+05	x	x	x	x
SMD11	1.49E+03	2.05E+05	4.89E+02	3.48E+05	9.34E+02	7.27E+05	1.14E+03	8.96E+05				
SMD12	1.19E+03	1.90E+05	4.48E+02	3.43E+05	1.06E+03	9.74E+05	6.42E+02	2.16E+05	x	x	x	x
	4.99E+02	*7.87E+04*										

problems, the non-linearity in the functions is more severe, for which quadratic approximations were not successful, and consequently the function evaluations are much higher. The function evaluations used by SABLA are of roughly similar

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TEVC.2017.2670659, IEEE Transactions on Evolutionary Computation



(a) BLTP1 (b) BLTP2 (c) BLTP3 (d) BLTP4

(e) BLTP5 (f) BLTP6 (g) BLTP7 (h) BLTP8

(i) BLTP9 (j) BLTP10 (k) BLTP11 (l) BLTP12

(m) BLTP13

Fig. 1. Median upper level convergence behavior using SABLA on BLTP series problems (Note: BLTP{4,7,9,12,13} are maximization problems)

order for most problems, since it treats each problem as a generic black-box problem and applies the surrogate building and re-evaluation steps in order to obtain accurate lower level optima. The advantage of doing so is evident from the results discussed in Table II.

Moreover, the experiments also suggest that SABLA is able to achieve low error values substantially before the run is completed. For the cases where a median run of SABLA was able to achieve the accuracy of 1×10^{-6} at *both* levels, the median function evaluations to reach the best accuracy are also shown in italics in Table III, which are substantially lower than the median final evaluations. The convergence plots for the median upper level runs for each problem are shown in Figure 1 for BLTP and Figure 2 for SMD problems. It can be observed from the plots that for almost all problems, SABLA converged to the near-final values within 20% of total (final) evaluations. For the cases where the convergence plot appears flat, the best solution was obtained in the first generation itself.

2) Performance Profile: In addition to the performance on individual problems above, we also present a visual statistical comparison of the performance using *performance profile*. Since the results from BIDE are not available for SMD7-SMD12, we present two sets of performance profiles. First one is based on 19 problems (BLTP1-BLTP13 and SMD1-SMD6) and includes BIDE in the set of compared algorithms (Figure 4), while the second one is based on all 25 problems and excluded BIDE (Figure 5). The function $\rho(\tau)$ is the cumulative distribution function for performance measure τ. The τ value

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TEVC.2017.2670659, IEEE Transactions on Evolutionary Computation

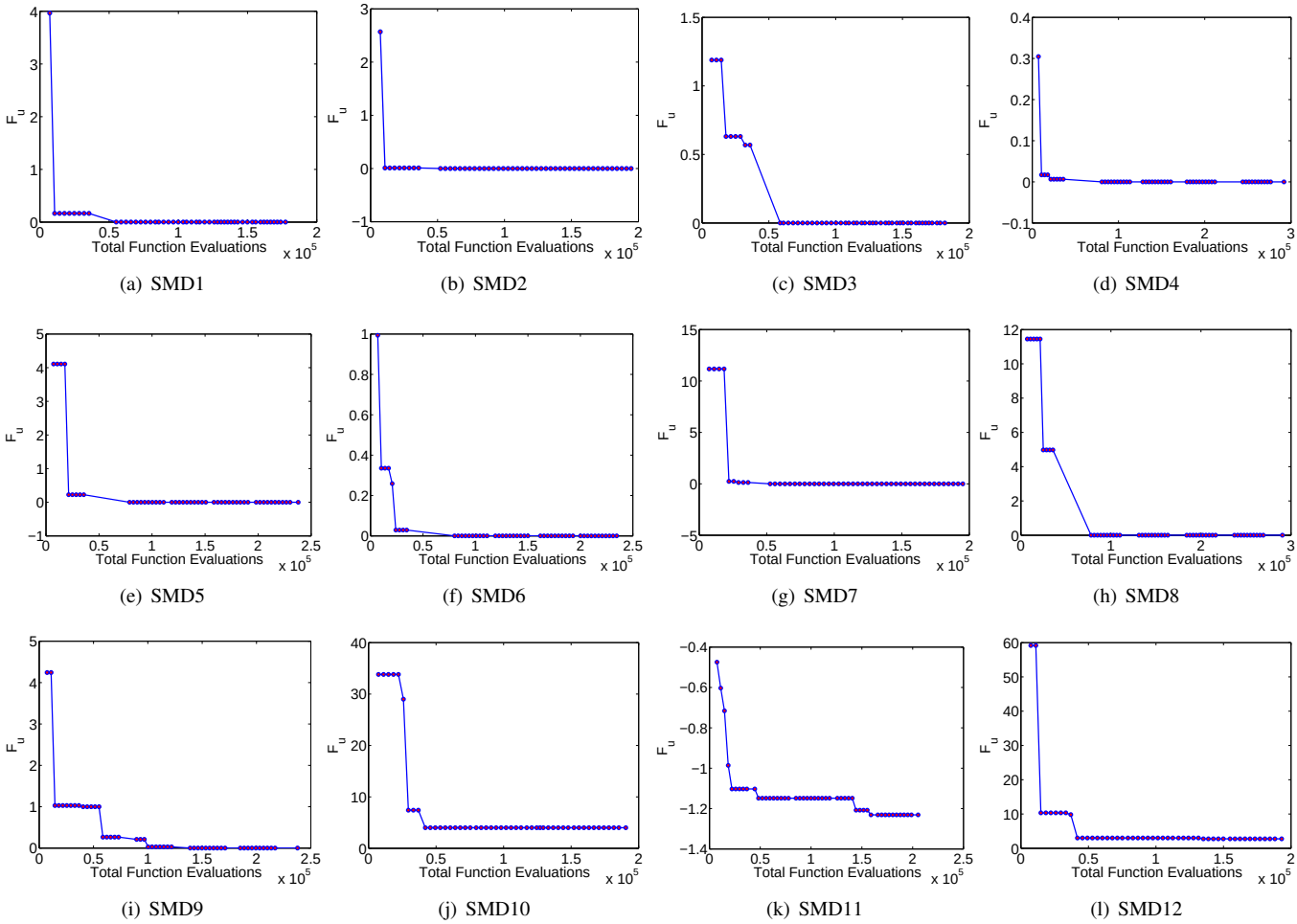


Fig. 2. Median upper level convergence behavior using SABLA on SMD series problems

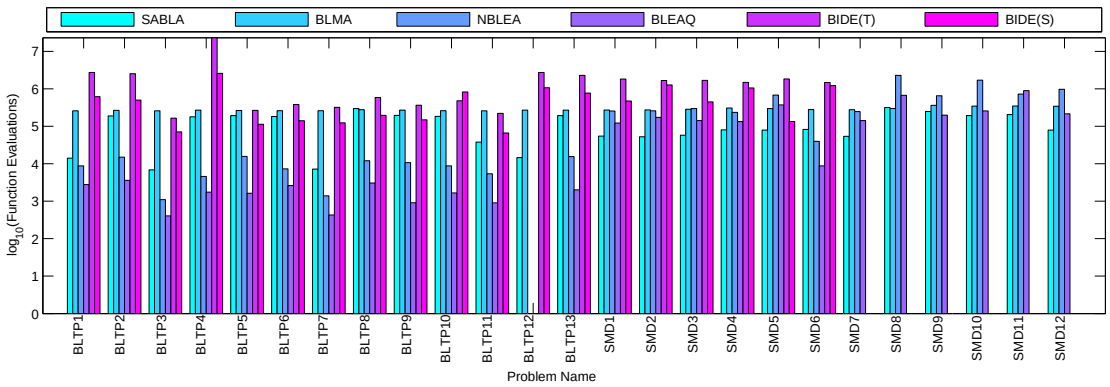
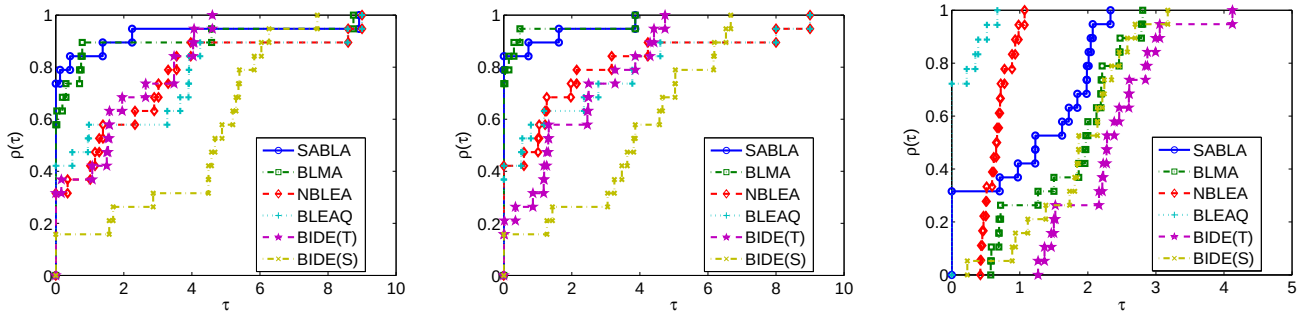


Fig. 3. Comparison of function evaluations(UL+LL) used by SABLA, BLMA, NBLEA, BLEAQ, BIDE(T) and BIDE(S) the SMD and BLTP benchmarks

effectively represents the factor (ratio) of the performance compared to the best performance. Thus the minimum value for τ is 1 (or 0 if plotted in log scale) which corresponds to the best result. In our study, the median accuracy (ε_u and ε_l) is used as the performance measure and τ is accordingly the ratio of median accuracy obtained by an algorithm to the best accuracy obtained for that problem, plotted in $\log_{10}$ scale. The performance profiles corresponding to the the total function evaluations used are generated in the similar way.

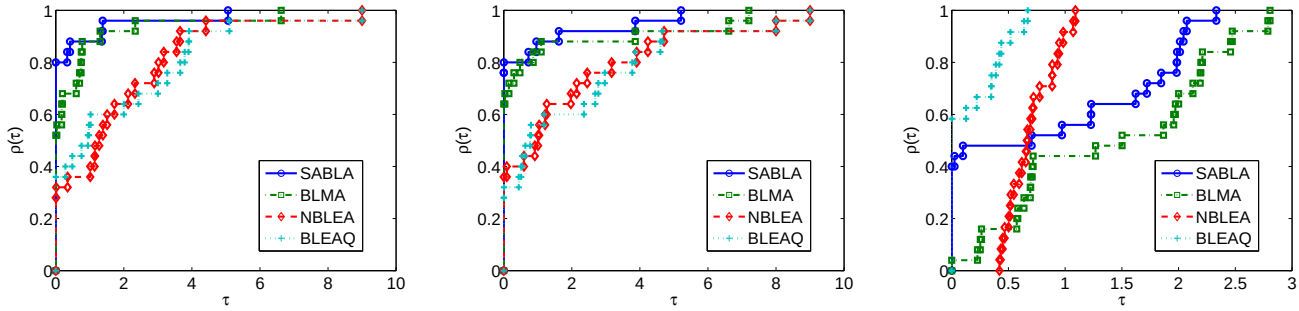
From the profiles in Figure 4(a)-4(b) it can be seen that for both upper and lower level accuracies, the profile of SABLA lies to the left of the others (with exception of slight overlap with BLMA), indicating its superior performance. NBLEA, BLEAQ and BIDE(T) have comparable performances to each other at both upper and lower levels. The performance of BIDE(S) is poorer than others at both levels. The trends are

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TEVC.2017.2670659, IEEE Transactions on Evolutionary Computation



(a) Upper level (b) Lower level (c) Total FE

Fig. 4. Performance profiles for median accuracy(ϵ_u, ϵ_l) and total function evaluations obtained using SABLA, BLMA, NBLEA, BLEAQ, BIDE(T) and BIDE(S) and for 19 bilevel test problems (exclude SMD7-SMD12)



(a) Upper level (b) Lower level (c) Total FE

Fig. 5. Performance profiles for median accuracy(ϵ_u, ϵ_l) and total function evaluations obtained using SABLA, BLMA, NBLEA and BLEAQ for 25 bilevel test problems

different in terms of number of function evaluations, as evident from Figure 4(c). For approximately 32% of the problems ($\rho \approx 0.32$), the number of function evaluations used by SABLA are the lowest, compared to BLEAQ which has the lowest value for approximately 72% problems ($\rho \approx 0.72$). SABLA performs overall worse than BLEAQ and better than BLMA, BIDE(T) and BIDE(S). With respect to NBLEA, it is better for approximately one-third of the problems ($\rho \approx 0.32$), and worse thereafter. BIDE(S) shows lower evaluations compared to BIDE(T) consistently.

The comparisons between SABLA, BLMA, NBLEA and BLEAQ on the full set of 25 problems show similar trends in Figure 5. SABLA performs the best in terms of accuracies, followed by BLMA, whereas NBLEA and BLMA performances are roughly equivalent. In terms of function evaluations, BLEAQ performs the best and BLMA the worst. SABLA performs better than NBLEA for approximately half of the problems ($\rho \approx 0.48$), and worse for the rest.

To summarize, it can be inferred from the numerical experiments that BIDE(T) and BIDE(S) used the highest number of function evaluations, but show relatively poor results. NBLEA and BLEAQ use low number of evaluations for linear/quadratic problems (BLTP), but higher number of evaluations for more non-linear (SMD) problems. SABLA is able to achieve best results for most of the problems. It nominally uses higher number of function evaluations than NBLEA/BLEAQ for BLTP and lower for SMD problems. Thus SABLA shows a significant potential for solving generic non-linear problems with reasonable computational expense.

E. Further experiments: effect of different components in SABLA

In the last set of experiments, we investigate the impact of different components incorporated within SABLA. In particular, the intent is to establish if having re-evaluation, nested local search and multiple surrogates show notable benefits over the range of problems studied.

First we compare the performance of SABLA with three different variations: one that doesn't use the nested local search, one that doesn't use the re-evaluations, and one that doesn't use either. In order to conduct this experiment, we increase the number of generations at the upper level to 80 to compensate for the evaluations that the default version uses for local search and/or re-evaluation. The results are summarized in Table IV. From the comparison, it is evident that removing either re-evaluation or nested local search is detrimental to the performance, and removing both deteriorates the performance even further. However, it is interesting to note that the impact of removing re-evaluation on performance is more prominent than the nested local search. The reason for this is because the role of re-evaluation in SABLA is to give the best possible estimate of an upper level evaluation, so that the upper level search is not misguided. It can be seen that when re-evaluation is in place (without nested local search), the algorithm is

able to at least reach in the vicinity of the optimum. The role of nested local search is to drive it further close to the optimum. Without re-evaluation, the nested local search is not able to attain this because the search gets misled. The behavior can be easily interpreted by observing the population for SMD10 problem when run without re-evaluation and nested local search. Consider two solutions in the final population from one of the runs. The first one has $\mathbf{x_u}$ = {-3.70016, -2.60948}, $\mathbf{x_l}$ = {-4.68393, -1.71463, -1.5608}, which gives F_u=-9405.65 and f_l=9556.428. However, for this $\mathbf{x_u}$, when a thorough lower level optimization (i.e., re-evaluation) is performed, it results in $\mathbf{x_l}$ = {0.795472, 0.91122, -1.23183}, F_u = 55.15079 and f_l = 16.37893. Another solution has $\mathbf{x_u}$={0.521878, 0.350444}, $\mathbf{x_l}$={-1.17377, -1.16184, -0.3453} with F_u=7.129151 and f_l=20.84668. When re-evaluated using an extensive lower level search, it results in F_u= 6.890112 and f_l=18.4591. Thus, it can be seen that without re-evaluation, the first solution will be ranked better than upper level, whereas with re-evaluation, the second solution will be ranked higher. Therefore the algorithm may be easily misled towards a wrong solution if the lower level optimum is inaccurate, which demonstrates the importance of re-evaluation.

Lastly, we observe the performance of SABLA using only a single surrogate (either RSM1, RSM2 or Kriging) instead of multiple. The results are shown in Table V. From the table, it can be observed that all strategies are able to reach close to the true optimum, and there are extremely small variations between them for most problems. Among the individual strategies, RSM2 performed the best for the set of problems considered here. In fact, its performance is at par to that the case with multiple strategies. However, in absence of prior knowledge about the functions, it cannot determined which model is likely to fit the observed data the best. Using multiple surrogates thus provides more flexibility and reliability to deal with generic functions, as also observed in some of the recent studies in the domain of surrogate assisted optimization [38].

IV. Summary and future work

Bilevel optimization is a problem of theoretical and practical interest, with a number of unique challenges when compared to traditional single level optimization problems. Considering some of these in strict form, the past efforts were directed in obtaining optimal solutions to linear/quadratic bilevel problems using exact classical approaches. The developments in recent years have been from a relatively more practical perspective, where the objective/constraint functions may not have required mathematical properties to apply such techniques. Consequently, the evolutionary and hybrid approaches have been increasingly used to solve such problems, although the optimality at lower level (and indeed upper level) cannot be guaranteed. The existing evolutionary/hybrid approaches tend to use high number of evaluations to obtain competitive solutions. Surrogate modeling is a promising approach which can be used to improve on the state-of-the-art. In this paper, we take a step in this direction by developing a surrogate assisted bilevel algorithm (SABLA). The proposed approach uses multiple surrogates to approximate the lower level objective/constraint functions, providing much more flexibility to model generic non-linear functions accurately. Three models – RSM1, RSM2 and Kriging – are used in the proposed study, and the set can be easily extended to include other models. Apart from the surrogate modeling, a number of other mechanisms are incorporated to improve the performance of the algorithm. These include a local search at upper level, periodic update of surrogate models at lower level, re-evaluation of promising solutions through a hybrid search and enhanced ranking process at upper level, where the re-evaluated solutions are preferred over others. Numerical experiments are conducted over a set of 25 standard problems used in the domain for performance assessment. Comparisons with prominent contemporary algorithms clearly show that the proposed approach is able to achieve the best or competitive results for most of the problems. At the same time, the number of function evaluations used by SABLA to achieve the results are lower than all others, except for linear/quadratic cases where NBLEA/BLMA use lower number of function evaluations.

Thus, overall the proposed approach is able to achieve a favorable balance between accuracy and computational expense for solving single-objective bilevel optimization problems. There is also a significant scope of further improvements in the method, for example use of other different type of models (e.g. neural networks, radial basis functions) within the framework, and surrogate modeling at both levels, in order to make the approach more viable for computationally expensive problems. Additionally, the use of surrogate assisted methods might become difficult itself as the dimensionality (number of variables) of the problem grows, since the training time as well as accuracy of certain surrogate models may be adversely affected. To develop strategies to tackle such high-dimensional cases would form another interesting research direction.

Acknowledgment

The second and third authors would like to acknowledge Early Career Researcher (ECR) grant from UNSW and Future Fellowship from the Australian Research Council, respectively. The authors would also like to thank Dr. Ankur Sinha for assistance with NBLEA and BLEAQ codes.

References

[1] C. Kirjner-Neto, E. Polak, and A. Der Kiureghian, "An outer approximations approach to reliability-based optimal design of structures," *Journal of optimization theory and applications*, vol. 98, no. 1, pp. 1–16, 1998.

[2] H. Sun, Z. Gao, and J. Wu, "A bi-level programming model and solution algorithm for the location of logistics distribution centers," *Applied mathematical modelling*, vol. 32, no. 4, pp. 610–616, 2008.

[3] A. Sinha, P. Malo, A. Frantsev, and K. Deb, "Multi-objective stackelberg game between a regulating authority and a mining company: A case study in environmental economics," in *IEEE Congress on Evolutionary Computation (CEC)*, 2013, pp. 478–485.

[4] A. Migdalas, "Bilevel programming in traffic planning: models, methods and challenge," *Journal of Global Optimization*, vol. 7, no. 4, pp. 381–405, 1995.

[5] R. G. Jeroslow, "The polynomial hierarchy and a simple model for competitive analysis," *Mathematical Programming*, vol. 32, no. 2, pp. 146–164, 1985. [Online]. Available: http://dx.doi.org/10.1007/BF01586088

TABLE IV
COMPARISON OF PERFORMANCE BETWEEN DIFFERENT VERSIONS OF SABLA

Problem Name	SABLA							
	Without nested Local search		Without re-evaluation		Without both		With both (default)	
	UL	LL	UL	LL	UL	LL	UL	LL
Accuracy								
BLTP1	**1.00E-06**	**1.00E-06**	**1.00E-06**	**1.00E-06**	**1.00E-06**	**1.00E-06**	**1.00E-06**	**1.00E-06**
BLTP2	4.30E-03	1.12E-02	5.55E-02	1.00E-06	5.89E-02	**1.00E-06**	**1.34E-06**	3.53E-06
BLTP3	**1.00E-06**	**1.00E-06**	**1.00E-06**	**1.00E-06**	**1.00E-06**	**1.00E-06**	**1.00E-06**	**1.00E-06**
BLTP4	**1.00E-06**	**1.00E-06**	**1.00E-06**	**1.00E-06**	**1.00E-06**	**1.00E-06**	8.21E+02	**1.00E-06**
BLTP5	7.34E-06	8.38E-05	7.41E-06	8.40E-05	7.41E-06	8.40E-05	**4.46E-06**	**7.77E-05**
BLTP6	**2.21E-06**	**1.26E-14**	7.07E-03	1.22E-04	1.56E-02	3.71E-04	1.75E-04	**1.00E-06**
BLTP7	**1.00E-06**	**1.00E-06**	**1.00E-06**	**1.00E-06**	**1.00E-06**	**1.00E-06**	**1.00E-06**	**1.00E-06**
BLTP8	**1.24E-05**	**1.00E-06**	2.12E-02	4.10E-04	2.21E-02	5.77E-04	3.42E-05	**1.00E-06**
BLTP9	8.06E-06	1.78E-05	9.09E-06	1.82E-05	9.09E-06	1.82E-05	**6.51E-06**	**1.72E-05**
BLTP10	**1.11E-06**	9.98E-04	2.02E-01	4.29E-01	2.17E-01	3.50E-01	2.63E-06	**4.65E-04**
BLTP11	**1.00E-06**	**1.00E-06**	**1.00E-06**	**1.00E-06**	**1.00E-06**	**1.00E-06**	**1.00E-06**	**1.00E-06**
BLTP12	**1.00E-06**	**1.00E-06**	2.67E-04	2.16E-03	**1.00E-06**	**1.00E-06**	**1.00E-06**	**1.00E-06**
BLTP13	3.00E-04	2.19E-05	**4.67E-05**	6.37E-06	5.95E-05	**6.35E-06**	6.01E-04	4.16E-05
SMD1	3.20E-03	3.20E-03	**1.00E-06**	**1.00E-06**	7.82E-03	5.87E-03	**1.00E-06**	**1.00E-06**
SMD2	8.94E-04	4.95E-04	9.48E+00	2.30E+01	9.51E+00	2.70E+01	**1.00E-06**	**1.00E-06**
SMD3	2.55E-02	2.53E-03	**1.00E-06**	**1.00E-06**	5.69E-02	3.28E-01	**1.00E-06**	**1.00E-06**
SDM4	3.14E-05	2.22E-05	1.55E+00	1.90E+00	1.33E+00	1.99E+00	**1.00E-06**	**1.00E-06**
SDM5	2.84E-02	6.84E-03	8.30E+01	8.89E+01	7.94E+01	9.00E+01	**1.00E-06**	**1.00E-06**
SMD6	1.00E-06	1.00E-06	**1.00E-06**	**1.00E-06**	**1.00E-06**	**1.00E-06**	**1.00E-06**	**1.00E-06**
SMD7	6.55E-03	1.06E-04	9.42E+00	3.86E+01	1.06E+01	3.86E+01	**1.00E-06**	**1.00E-06**
SMD8	7.57E-01	5.05E-02	2.37E+02	2.68E+02	2.52E+02	2.71E+02	**2.76E-03**	**6.77E-04**
SMD9	6.89E-03	4.87E-03	2.50E+01	3.17E+01	2.45E+01	3.01E+01	**1.78E-06**	**1.77E-06**
SMD10	2.21E+00	4.88E-01	1.05E+04	1.07E+04	1.05E+04	1.06E+04	**3.80E-06**	**3.90E-06**
SMD11	**2.08E-01**	**2.50E-01**	2.00E+02	2.00E+02	2.00E+02	2.00E+02	2.31E-01	3.25E-01
SMD12	8.48E-02	1.50E+00	1.14E+02	2.07E+02	1.07E+02	1.99E+02	**1.00E-06**	**1.00E-06**
Function evaluations								
BLTP1	1.05E+02	1.76E+04	2.00E+02	2.40E+04	6.00E+01	7.20E+03	8.40E+01	1.40E+04
BLTP2	1.70E+03	2.88E+05	2.34E+03	2.30E+05	1.62E+03	1.94E+05	1.27E+03	1.89E+05
BLTP3	4.20E+01	6.86E+03	1.20E+02	1.44E+04	4.00E+01	4.80E+03	4.20E+01	6.86E+03
BLTP4	4.20E+01	6.99E+03	2.44E+02	2.71E+04	6.00E+01	7.20E+03	1.17E+03	1.78E+05
BLTP5	1.70E+03	2.92E+05	1.88E+03	2.04E+05	1.62E+03	1.94E+05	1.28E+03	1.92E+05
BLTP6	1.70E+03	2.80E+05	1.80E+03	1.98E+05	1.62E+03	1.94E+05	1.26E+03	1.82E+05
BLTP7	4.20E+01	7.16E+03	3.21E+02	2.91E+04	6.00E+01	7.20E+03	4.20E+01	7.18E+03
BLTP8	1.70E+03	2.84E+05	3.27E+03	3.85E+05	1.62E+03	1.94E+05	2.11E+03	2.95E+05
BLTP9	1.70E+03	2.87E+05	1.93E+03	2.12E+05	1.62E+03	1.94E+05	1.29E+03	1.95E+05
BLTP10	1.70E+03	2.83E+05	2.71E+03	2.03E+05	1.62E+03	1.94E+05	1.69E+03	1.83E+05
BLTP11	3.15E+02	5.13E+04	2.31E+02	2.64E+04	2.80E+02	3.36E+04	2.41E+02	3.77E+04
BLTP12	1.26E+02	2.20E+04	1.92E+03	2.02E+05	1.60E+02	1.92E+04	8.40E+01	1.45E+04
BLTP13	1.70E+03	2.93E+05	1.85E+03	2.09E+05	1.62E+03	1.94E+05	1.15E+03	1.93E+05
SMD1	1.66E+03	2.40E+05	3.32E+02	4.29E+04	1.62E+03	1.94E+05	3.42E+02	5.47E+04
SMD2	1.70E+03	2.87E+05	2.81E+03	3.07E+05	1.62E+03	1.94E+05	3.54E+02	5.24E+04
SMD3	1.66E+03	2.41E+05	1.08E+03	1.52E+05	1.62E+03	1.94E+05	3.42E+02	5.78E+04
SDM4	1.70E+03	2.89E+05	3.13E+03	4.58E+05	1.62E+03	1.94E+05	4.82E+02	7.99E+04
SDM5	1.70E+03	2.86E+05	3.63E+03	4.57E+05	1.62E+03	1.94E+05	4.82E+02	7.92E+04
SMD6	1.28E+03	2.09E+05	3.88E+02	5.56E+04	7.80E+02	9.36E+04	4.82E+02	8.22E+04
SMD7	1.70E+03	2.87E+05	3.12E+03	3.74E+05	1.62E+03	1.94E+05	3.54E+02	5.38E+04
SMD8	1.70E+03	2.87E+05	3.62E+03	6.35E+05	1.62E+03	1.94E+05	2.33E+03	3.15E+05
SMD9	1.70E+03	2.88E+05	3.00E+03	2.84E+05	1.62E+03	1.94E+05	1.61E+03	2.48E+05
SMD10	1.70E+03	2.96E+05	2.02E+03	2.11E+05	1.62E+03	1.94E+05	1.21E+03	1.92E+05
SMD11	1.70E+03	2.96E+05	2.28E+03	2.23E+05	1.62E+03	1.94E+05	1.49E+03	2.05E+05
SMD12	1.70E+03	2.95E+05	2.09E+03	2.10E+05	1.62E+03	1.94E+05	4.99E+02	7.87E+04

[6] R. Cassidy, M. Kirby, and W. Raike, "Efficient distribution of resources through three levels of government," *Management Science*, vol. 17, no. 8, pp. B–462, 1971.

[7] J. Bracken and J. T. McGill, "Mathematical programs with optimization problems in the constraints," *Operations Research*, vol. 21, no. 1, pp. 37–44, 1973.

[8] H. Von Stackelberg, *The theory of the market economy*. Oxford University Press, 1952.

[9] L. N. Vicente and P. H. Calamai, "Bilevel and multilevel programming: A bibliography review," *Journal of Global Optimization*, vol. 5, no. 3, pp. 291–306, 1994.

[10] B. Colson, P. Marcotte, and G. Savard, "An overview of bilevel optimization," *Annals of operations research*, vol. 153, no. 1, pp. 235–256, 2007.

[11] J. S. Angelo and H. J. C. Barbosa, "A study on the use of heuristics to solve a bilevel programming problem," *International Transactions in Operational Research*, vol. 22, no. 5, pp. 861–882, 2015. [Online]. Available: http://dx.doi.org/10.1111/itor.12153

[12] A. Sinha, P. Malo, and K. Deb, "Efficient evolutionary algorithm for single-objective bilevel optimization," *arXiv preprint arXiv:1303.3901*, 2013.

[13] J. Branke and K. Lu, "Finding the trade-off between robustness and worst-case quality," in *Proceedings of the 2015 Annual Conference on Genetic and Evolutionary Computation*. ACM, 2015, pp. 623–630.

[14] K. Lu, J. Branke, and T. Ray, "Improving efficiency of bi-level worst case optimization," in *International Conference on Parallel Problem Solving from Nature*. Springer, 2016, pp. 410–420.

[15] S. Dempe, "A simple algorithm for the-linear bilevel programming problem," *Optimization*, vol. 18, no. 3, pp. 373–385, 1987.

[16] H. Önal, "A modified simplex approach for solving bilevel linear programming problems," *European Journal of Operational Research*, vol. 67, no. 1, pp. 126–135, 1993.

[17] J. F. Bard and J. E. Falk, "An explicit solution to the multi-level programming problem," *Computers & Operations Research*, vol. 9, no. 1, pp. 77–100, 1982.

[18] L. Vicente, G. Savard, and J. Júdice, "Descent approaches for quadratic bilevel programming," *Journal of Optimization Theory and Applications*, vol. 81, no. 2, pp. 379–399, 1994.

[19] D. J. White and G. Anandalingam, "A penalty function approach for

TABLE V
RESULTS USING SABLA WITH SINGLE SURROGATES (RSM1, RSM2 OR KRIGING)

Prb No	RSM1		RSM2		Kriging		All (default)	
	UL	LL	UL	LL	UL	LL	UL	LL
Accuracy								
BLTP1	**1.00E-06**	**1.00E-06**	**1.00E-06**	**1.00E-06**	**1.00E-06**	**1.00E-06**	**1.00E-06**	**1.00E-06**
BLTP2	**1.34E-06**	**3.53E-06**	**1.34E-06**	**3.53E-06**	1.53E-06	4.02E-06	**1.34E-06**	**3.53E-06**
BLTP3	**1.00E-06**	**1.00E-06**	**1.00E-06**	**1.00E-06**	**1.00E-06**	**1.00E-06**	**1.00E-06**	**1.00E-06**
BLTP4	6.76E+02	**1.00E-06**	7.82E+02	**1.00E-06**	**2.80E+02**	**1.00E-06**	8.21E+02	**1.00E-06**
BLTP5	6.27E-06	8.15E-05	**4.46E-06**	**7.77E-05**	5.00E-06	7.89E-05	**4.46E-06**	**7.77E-05**
BLTP6	2.12E-02	1.12E-06	**1.75E-04**	**1.00E-06**	1.74E-04	1.00E-06	**1.75E-04**	1.00E-06
BLTP7	**1.00E-06**	**1.00E-06**	**1.00E-06**	**1.00E-06**	1.21E-05	8.65E-06	**1.00E-06**	**1.00E-06**
BLTP8	4.65E-05	**1.00E-06**	3.99E-05	**1.00E-06**	4.65E-05	**1.00E-06**	**3.42E-05**	**1.00E-06**
BLTP9	6.51E-06	**1.72E-05**	6.51E-06	**1.72E-05**	**6.50E-06**	1.73E-05	6.51E-06	**1.72E-05**
BLTP10	**1.00E-06**	**1.88E-05**	2.63E-06	4.36E-04	1.72E-06	1.05E-04	2.63E-06	4.65E-04
BLTP11	**1.00E-06**	**1.00E-06**	**1.00E-06**	**1.00E-06**	**1.00E-06**	**1.00E-06**	**1.00E-06**	**1.00E-06**
BLTP12	**1.00E-06**	**1.00E-06**	**1.00E-06**	**1.00E-06**	**1.00E-06**	**1.00E-06**	**1.00E-06**	**1.00E-06**
BLTP13	2.58E-04	1.92E-05	**2.19E-04**	**7.35E-06**	3.32E-04	3.23E-05	6.01E-04	4.16E-05
SMD1	**1.00E-06**	**1.00E-06**	**1.00E-06**	**1.00E-06**	**1.00E-06**	**1.00E-06**	**1.00E-06**	**1.00E-06**
SMD2	**1.00E-06**	**1.00E-06**	**1.00E-06**	**1.00E-06**	**1.00E-06**	**1.00E-06**	**1.00E-06**	**1.00E-06**
SMD3	**1.00E-06**	**1.00E-06**	**1.00E-06**	**1.00E-06**	**1.00E-06**	**1.00E-06**	**1.00E-06**	**1.00E-06**
SDM4	**1.00E-06**	**1.00E-06**	**1.00E-06**	**1.00E-06**	**1.00E-06**	**1.00E-06**	**1.00E-06**	**1.00E-06**
SDM5	**1.00E-06**	**1.00E-06**	**1.00E-06**	**1.00E-06**	**1.00E-06**	**1.00E-06**	**1.00E-06**	**1.00E-06**
SMD6	**1.00E-06**	**1.00E-06**	**1.00E-06**	**1.00E-06**	**1.00E-06**	**1.00E-06**	**1.00E-06**	**1.00E-06**
SMD7	**1.00E-06**	**1.00E-06**	**1.00E-06**	**1.00E-06**	**1.00E-06**	**1.00E-06**	**1.00E-06**	**1.00E-06**
SMD8	2.85E-03	6.99E-04	**2.16E-03**	**5.31E-04**	2.54E-03	6.23E-04	2.76E-03	6.77E-04
SMD9	2.00E-06	1.97E-06	**1.13E-06**	**1.07E-06**	1.96E-06	1.84E-06	1.78E-06	1.77E-06
SMD10	1.60E+01	4.03E-06	3.96E-06	3.90E-06	8.10E-02	4.03E-06	**3.80E-06**	**3.90E-06**
SMD11	**1.87E-01**	4.66E-01	2.40E-01	3.64E-01	2.72E-01	3.65E-01	2.31E-01	**3.25E-01**
SMD12	**1.00E-06**	**1.00E-06**	**1.00E-06**	**1.00E-06**	**1.00E-06**	**1.00E-06**	**1.00E-06**	**1.00E-06**
Function evaluations								
BLTP1	8.40E+01	1.39E+04	8.40E+01	1.40E+04	8.40E+01	1.40E+04	8.40E+01	1.40E+04
BLTP2	1.27E+03	1.89E+05	1.27E+03	1.89E+05	1.27E+03	1.90E+05	1.27E+03	1.89E+05
BLTP3	4.20E+01	6.86E+03	4.20E+01	6.86E+03	4.20E+01	6.86E+03	4.20E+01	6.86E+03
BLTP4	1.17E+03	1.78E+05	1.17E+03	1.79E+05	1.17E+03	1.78E+05	1.17E+03	1.78E+05
BLTP5	1.27E+03	1.92E+05	1.28E+03	1.92E+05	1.27E+03	1.92E+05	1.28E+03	1.92E+05
BLTP6	1.28E+03	1.79E+05	1.26E+03	1.82E+05	1.29E+03	1.82E+05	1.26E+03	1.82E+05
BLTP7	6.30E+01	1.06E+04	6.30E+01	1.06E+04	1.12E+03	1.83E+05	4.20E+01	7.18E+03
BLTP8	2.04E+03	2.90E+05	2.11E+03	3.01E+05	2.11E+03	3.00E+05	2.11E+03	2.95E+05
BLTP9	1.29E+03	1.95E+05	1.29E+03	1.95E+05	1.29E+03	1.95E+05	1.29E+03	1.95E+05
BLTP10	1.67E+03	1.82E+05	1.71E+03	1.83E+05	1.64E+03	1.85E+05	1.69E+03	1.83E+05
BLTP11	2.41E+02	3.77E+04	2.41E+02	3.77E+04	2.41E+02	3.77E+04	2.41E+02	3.77E+04
BLTP12	8.40E+01	1.47E+04	6.30E+01	1.10E+04	8.40E+01	1.47E+04	8.40E+01	1.45E+04
BLTP13	1.19E+03	1.95E+05	1.19E+03	1.94E+05	1.16E+03	1.93E+05	1.15E+03	1.93E+05
SMD1	3.42E+02	5.37E+04	3.42E+02	5.47E+04	3.42E+02	5.47E+04	3.42E+02	5.47E+04
SMD2	3.54E+02	5.86E+04	3.54E+02	5.24E+04	3.54E+02	5.23E+04	3.54E+02	5.24E+04
SMD3	3.42E+02	5.73E+04	3.42E+02	5.69E+04	3.42E+02	5.77E+04	3.42E+02	5.78E+04
SDM4	4.82E+02	8.09E+04	4.82E+02	7.94E+04	4.82E+02	7.96E+04	4.82E+02	7.99E+04
SDM5	4.82E+02	8.05E+04	4.82E+02	7.80E+04	4.82E+02	8.17E+04	4.82E+02	7.92E+04
SMD6	4.82E+02	7.89E+04	4.14E+02	6.90E+04	4.58E+02	8.01E+04	4.82E+02	8.22E+04
SMD7	7.88E+02	1.31E+05	3.54E+02	5.28E+04	3.54E+02	5.34E+04	3.54E+02	5.38E+04
SMD8	2.33E+03	2.97E+05	2.33E+03	2.96E+05	2.33E+03	3.01E+05	2.33E+03	3.15E+05
SMD9	1.61E+03	2.48E+05	1.54E+03	2.42E+05	1.60E+03	2.53E+05	1.61E+03	2.48E+05
SMD10	1.19E+03	1.91E+05	1.19E+03	1.91E+05	1.19E+03	1.91E+05	1.21E+03	1.92E+05
SMD11	1.43E+03	2.02E+05	1.45E+03	2.02E+05	1.50E+03	2.05E+05	1.49E+03	2.05E+05
SMD12	9.47E+02	1.22E+05	5.03E+02	7.87E+04	4.94E+02	7.85E+04	4.99E+02	7.87E+04

solving bi-level linear programs," *Journal of Global Optimization*, vol. 3, no. 4, pp. 397–419, 1993.

[20] E. Aiyoshi and K. Shimizu, "A solution method for the static constrained Stackelberg problem via penalty method," *Automatic Control, IEEE Transactions on*, vol. 29, no. 12, pp. 1111–1114, 1984.

[21] R. Mathieu, L. Pittard, and G. Anandalingam, "Genetic algorithm based approach to bi-level linear programming," *RAIRO. Recherche Opérationnelle*, vol. 28, no. 1, pp. 1–21, 1994.

[22] J. F. Bard, "An efficient point algorithm for a linear two-stage optimization problem," *Operations Research*, vol. 31, no. 4, pp. 670–684, 1983.

[23] J. Rajesh, K. Gupta, H. S. 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*, vol. 9, no. 4, pp. 307–319, 2003.

[24] S. R. Hejazi, A. Memariani, G. Jahanshahloo, and M. M. Sepehri, "Linear bilevel programming solution by genetic algorithm," *Computers & Operations Research*, vol. 29, no. 13, pp. 1913–1925, 2002.

[25] V. Oduguwa and R. Roy, "Bi-level optimisation using genetic algorithm," in *Artificial Intelligence Systems, 2002.(ICAIS 2002). IEEE International Conference on*, 2002, pp. 322–327.

[26] D. Wang and G. Du, "A self adaptive penalty function based genetic algorithm for value-bilevel programming problem," *International Journal on Computer Science and Engineering*, vol. 3, no. 2, pp. 136–146, 2014.

[27] R. Kuo and C. Huang, "Application of particle swarm optimization algorithm for solving bi-level linear programming problem," *Computers & Mathematics with Applications*, vol. 58, no. 4, pp. 678–685, 2009.

[28] W. Halter and S. Mostaghim, "Bilevel optimization of multi-component chemical systems using particle swarm optimization," in *IEEE Congress on Evolutionary Computation (CEC)*, 2006, pp. 1240–1247.

[29] A. Sinha, P. Malo, and K. Deb, "Test problem construction for single-objective bilevel optimization," *Evolutionary computation*, vol. 22, no. 3, pp. 439–477, 2014.

[30] ——, "An improved bilevel evolutionary algorithm based on quadratic approximations," in *IEEE Congress on Evolutionary Computation (CEC)*, July 2014, pp. 1870–1877.

[31] J. S. Angelo, E. Krempser, and H. J. Barbosa, "Differential evolution for bilevel programming," in *IEEE Congress on Evolutionary Computation (CEC)*, 2013, pp. 470–477.

[32] X. Zhu, Q. Yu, and X. Wang, "A hybrid differential evolution algorithm for solving nonlinear bilevel programming with linear constraints,"

in Proceedings of 5th IEEE International Conference on Cognitive Informatics (ICCI06), pp. 126–131, 2006.
[33] A. Koh, "A metaheuristic framework for bi-level programming problems with multi-disciplinary applications," in *Metaheuristics for Bi-level Optimization*. Springer, 2013, pp. 153–187.
[34] M. M. Islam, H. K. Singh, and T. Ray, "A memetic algorithm for solving single objective bilevel optimization problems," in *IEEE Congress on Evolutionary Computation (CEC)*, 2015, pp. 1643–1650.
[35] M. M. Islam, H. K. Singh, T. Ray, and A. Sinha, "An enhanced memetic algorithm for single-objective bilevel optimization problems," *Evolutionary Computation*, 2016.
[36] G. G. Wang and S. Shan, "Review of metamodeling techniques in support of engineering design optimization," *Journal of Mechanical Design*, vol. 129, pp. 370–380, 2007. [Online]. Available: http://link.aip.org/link/?JMD/129/370/1
[37] Y. Jin, "Surrogate-assisted evolutionary computation: Recent advances and future challenges," *Swarm and Evolutionary Computation*, vol. 1, no. 2, pp. 61–70, 2011.
[38] K. S. Bhattacharjee, H. K. Singh, and T. Ray, "Multi-objective optimization with multiple spatially distributed surrogates," *Journal of Mechanical Design*, vol. 138, no. 9, p. 091401, 2016.
[39] D. Lim, Y. Jin, Y.-S. Ong, and B. Sendhoff, "Generalizing surrogate-assisted evolutionary computation," *IEEE Transactions on Evolutionary Computation*, vol. 14, no. 3, pp. 329–355, 2010.
[40] C. Sun, Y. Jin, J. Zeng, and Y. Yu, "A two-layer surrogate-assisted particle swarm optimization algorithm," *Soft Computing*, vol. 19, no. 6, pp. 1461–1475, 2015.
[41] Y. Jin, "A comprehensive survey of fitness approximation in evolutionary computation," *Soft Computing - A Fusion of Foundations, Methodologies and Applications*, vol. 9, no. 1, pp. 3–12, 2005.
[42] A. Sinha, P. Malo, and K. Deb, "Nested blievel evolutionary algorithm (N-BLEA) code and user guide," http:www.bilevel.org.
[43] ——, "Solving optimistic bilevel programs by iteratively approximating lower level optimal value function," in *IEEE Congress on Evolutionary Computation (CEC)*, 2016, pp. 1877–1884.
[44] J. S. Angelo, E. Krempser, and H. J. Barbosa, "Differential evolution assisted by a surrogate model for bilevel programming problems," in *IEEE Congress on Evolutionary Computation (CEC)*, 2014, pp. 1784–1791.
[45] A. Isaacs, "Development of optimization methods to solve computationally expensive problems," Ph.D. dissertation, The University of New South Wales, Australian Defence Force Academy, Canberra, Australia, 2009.
[46] T. Takahama and S. Sakai, "Constrained optimization by the ε constrained differential evolution with gradient-based mutation and feasible elites," in *IEEE Congress on Evolutionary Computation (CEC)*, 2006, pp. 1–8.
[47] W. Gong, Z. Cai, C. X. Ling, and C. Li, "Enhanced differential evolution with adaptive strategies for numerical optimization," *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, vol. 41, no. 2, pp. 397–413, 2011.
[48] R. H. Byrd, M. E. Hribar, and J. Nocedal, "An interior point algorithm for large-scale nonlinear programming," *SIAM Journal on Optimization*, vol. 9, no. 4, pp. 877–900, 1999.
[49] R. H. Byrd, J. C. Gilbert, and J. Nocedal, "A trust region method based on interior point techniques for nonlinear programming," *Mathematical Programming*, vol. 89, no. 1, pp. 149–185, 2000.
[50] R. H. Myers and D. C. Montgomery, *Response Surface Methodology: Process and Product in Optimzation using Designed Experiments*. John Wiley & Sons, Inc., NY, USA, 1995.
[51] J. Sacks, W. J. Welch, T. J. Mitchell, and H. P. Wynn, "Design and analysis of computer experiments," *Statistical science*, pp. 409–423, 1989.
[52] N. V. Queipo, J. V. Goicochea, and S. Pintos, "Surrogate modeling-based optimization of sagd processes," *Journal of Petroleum Science and Engineering*, vol. 35, no. 1, pp. 83–93, 2002.
[53] E. D. Dolan and J. J. Moré, "Benchmarking optimization software with performance profiles," *Mathematical programming*, vol. 91, no. 2, pp. 201–213, 2002.
[54] H. J. Barbosa, H. S. Bernardino, and A. Barreto, "Using performance profiles to analyze the results of the 2006 cec constrained optimization competition," in *IEEE Congress on Evolutionary Computation (CEC)*, 2010, pp. 1–8.
[55] J. F. Bard, "Convex two-level optimization," *Mathematical Programming*, vol. 40, no. 1-3, pp. 15–27, 1988.
[56] G. Savard and J. Gauvin, "The steepest descent direction for the nonlinear bilevel programming problem," *Operations Research Letters*, vol. 15, no. 5, pp. 265–272, 1994.
[57] G. Anandalingam and D. White, "A solution method for the linear static Stackelberg problem using penalty functions," *Automatic Control, IEEE Transactions on*, vol. 35, no. 10, pp. 1170–1173, 1990.
[58] J. E. Falk and J. Liu, "On bilevel programming, part i: general nonlinear cases," *Mathematical Programming*, vol. 70, no. 1-3, pp. 47–72, 1995.
[59] W. Candler and R. Townsley, "A linear two-level programming problem," *Computers & Operations Research*, vol. 9, no. 1, pp. 59–76, 1982.

Md Monjurul Islam is a PhD student at the School of Engineering and Information Technology, The University of New South Wales, Australia. He received his B.Sc. degree in Computer Science and Engineering from the Chittagong University of Engineering and Technology, Chittagong, Bangladesh. His primary research interest is development of computationally efficient techniques for solving bilevel optimization problems.

Hemant Kumar Singh completed his PhD from The University of New South Wales (UNSW) Australia in 2011 and B.Tech in Mechanical Engineering from the Indian Institute of Technology Kanpur in 2007. He is currently a lecturer in the School of Engineering and Information Technology, UNSW Australia. Prior to the current appointment, he has an experience of working with General Electric Aviation as a Lead Engineer for two years. His research interests include evolutionary computation and design optimization.

Tapabrata Ray is a Professor with the School of Engineering and Information Technology, The University of New South Wales, Australia. He leads the Multidisciplinary Design Optimization Group. His research interests include surrogate assisted/multi-fidelity optimization, multi/many-objective optimization, bilevel optimization, robust design and decision-making.