Solving Dual Problems Using a Coevolutionary Optimization Algorithm

Kalyanmoy Deb, Shivam Gupta, Joydeep Dutta, and Bhoomija Ranjan
Kanpur Genetic Algorithms Laboratory (KanGAL)
Indian Institute of Technology Kanpur
Kanpur, PIN 208016, India
Email: deb@iitk.ac.in
http://www.iitk.ac.in/kangal/pub.htm

KanGAL Report Number 2011009

Abstract

In solving certain optimization problems, the corresponding Lagrangian dual problem is sometimes solved simply because in these problems the dual problem is easier to solve than the original primal problem. Another reason for their solution is the implication of the weak duality theorem which suggests that for all optimization problems the optimal dual function value is smaller than or equal to the optimal primal objective value. The dual problem is a special case of a bilevel programming problem involving Lagrange multipliers as upper-level variables and decision variables as lower-level variables. In this paper, we propose a coevolutionary dual optimization (CEDO) algorithm for co-evolving two populations – one involving Lagrange multipliers and other involving decision variables – to find the dual solution. On 11 test problems taken from the optimization literature, we demonstrate the efficacy of CEDO algorithm by comparing it with a couple of nested algorithms and a couple of previously suggested coevolutionary algorithms. As a by-product, we analyze the test problems to find their associated duality gap and classify them into three categories having zero, finite or infinite duality gaps. The development of a coevolutionary approach, revealing the presence or absence of duality gap in a number of commonly-used test problems, and efficacy of the proposed algorithm compared to usual nested and other coevolutionary approaches remain as the hallmark of the current study.

1 Introduction

An optimization problem can be solved in two forms depending on the ease of solving one form over the other. In the most popular form, the primal problem in its original formulation is attempted to solve. An alternative approach is to convert the problem into an equivalent dual problem and solve. The variables in the dual problem are the Lagrange multipliers corresponding to constraints of the original problem. Thus, in some optimization problems, such as in linear programming problems having a comparatively smaller number of constraints than variables, the dual problem may be relatively easier to handle [7]. Dual problems are also meaningful to solve if the optimal solution of the dual problem can be used to construct the optimal solution of the original (or, primal) problem. It has been established in the optimization literature that the equivalence of primal and dual problems is related to a parameter called the *duality gap* – the difference between optimal objective values of the primal and the dual problems.

The dual formulation requires a nested optimization task which in most cases having non-linearities and high dimensionality in the primal problem, is computationally demanding. Moreover, the upper-level objective function of the dual problem formulation is usually non-differentiable at the dual solution, thereby making the task of finding dual solution using a classical derivative-based approach difficult. Evolutionary algorithms, with their population based approach and without the need of any derivative information, have been proved to be useful tools in handling such complexities in the past. Moreover, the existence of two interlinked sets of variables (Lagrange multipliers and decision variables) in the formulation of the dual problem makes the possibility of using a *coevolutionary* approach in solving the problem. Based on the importance of solving dual problems and the need for efficient solution methodologies, here, we propose a coevolutionary algorithm with two populations dedicated to performing separate yet interdependent optimization tasks. Our implementation is different from a couple of existing coevolutionary algorithms suggested for solving min-max problems [2, 3, 24] in that a classical optimization algorithm is deployed as a lower-level subproblem solver sparingly to help reduce evaluation noise associated with solving such problems. In this sense, the proposed method is hybrid in nature and takes advantage of coevolving two intertwined populations of primal and dual variables and at the same time invokes a classical optimization algorithm as a local search operator to improve the iterates.

The paper is organized in several sections. Section 2 gives a brief introduction on the theory of primal-dual problem formulation. The next section highlights some past studies attempted by other researchers in solving the dual problem. One of the common and simple-minded approaches is the so-called *nested* approach, in which, for a upper-level point the corresponding lower-level optimization problem is solved to local optimality using a classical optimization algorithm. Thereafter, past coevolutionary algorithms are reviewed and a couple of coevolutionary genetic algorithms (CGAs) are described in somewhat more details. Section 4 describes the proposed coevolutionary optimization algorithm highlighting the working principles of its operators. Section 5 presents the test problems used in this paper for demonstrating the working principle of the proposed coevolutionary procedure. Section 6 compiles extensive results obtained from solving 11 test problems using the proposed methodology, the nested procedures, and the existing coevolutionary approaches. Results are classified according to the duality gap associated with the problems. For some commonly-used test problems taken from the evolutionary algorithm literature, an analysis reveals some interesting conclusions between duality gap and performance of existing optimization algorithms. Based on the obtained results, conclusions are drawn in Section 7.

2 Primal-Dual Problems

An optimization problem statement can be written in the following form where $f:\ \Re^n \rightarrow \Re$, $\mathbf{g}$: $\Re^n \rightarrow \Re^m$:

$$\begin{aligned} \text{minimize} \quad & f(\mathbf{x}), \\ \text{subject to} \quad & \mathbf{g}(\mathbf{x}) \leq 0, \\ & \mathbf{x} \in X. \end{aligned} \tag{1}$$

Equation 1 represents, in general, the *primal problem* (P). For simplicity, equality constraints are not considered here, however with some modifications to our discussions here, they can be included as well. Let us denote the optimal solution to problem P be $\mathbf{x}^P$ and the corresponding function value as $f(\mathbf{x}^P)$, or simply f^P. The corresponding vector denoting a Lagrange multiplier for each constraint is represented by $\boldsymbol{\lambda}^P$ here.

The Lagrange function for the problem above can be written as:

$$L(\mathbf{x}, \boldsymbol{\lambda}) = f(\mathbf{x}) + \sum_{i=1}^{m} \lambda_i g_i(\mathbf{x}), \tag{2}$$

where λ_i is the dual variable corresponding to the i-th constraint. The *Lagrangian Dual Problem* (D) can then be written as follows [5]:

$$\begin{aligned} \text{maximize} \quad & \theta(\boldsymbol{\lambda}), \\ \text{subject to} \quad & \boldsymbol{\lambda} \in \Re^m_+, \\ \text{where} \quad & \theta(\boldsymbol{\lambda}) \quad = \inf\{L(\mathbf{x}, \boldsymbol{\lambda}) : \mathbf{x} \in X\} \\ & \qquad\quad = \inf\left\{ f(\mathbf{x}) + \sum_{i=1}^{m} \lambda_i g_i(\mathbf{x}) : \mathbf{x} \in X \right\}. \end{aligned} \tag{3}$$

For the dual problem D, components of $\boldsymbol{\lambda}$-vector are treated as variables. For an inequality constraint, λ_i is restricted to be non-negative. The function $\theta(\boldsymbol{\lambda})$, defined above, is known as the Lagrangian dual function. The dual formulation requires to find the infimum of Lagrange function $L(\mathbf{x}, \boldsymbol{\lambda})$, given in Equation 2, with respect to $\mathbf{x}$. The task of finding the infimum of the Lagrange function is a part of the task of solving the Lagrangian dual problem and is sometimes referred to as *Lagrangian dual subproblem*. To make our discussions easier, we shall drop the term 'Lagrangian' and simply refer problem D is as the dual problem and problem of finding $\theta(\boldsymbol{\lambda})$ as the dual subproblem. Let us denote the optimal solution to dual problem D as $\boldsymbol{\lambda}^D$. The corresponding $\mathbf{x}$ that is found to be the infimum solution to the dual subproblem is denoted by

$\mathbf{x}^D$. It is important here to highlight that there can be multiple infimum solutions ($\mathbf{x}^D$) to the subproblem or positive or negative infinity values to the components of $\mathbf{x}^D$. Importantly, $\mathbf{x}^D$ need not be feasible to the primal problem nor it be always identical to the primal solution $\mathbf{x}^P$. Let us also denote the optimal dual function value as $\theta(\boldsymbol{\lambda}^D)$ or simply θ^D.

We now discus the relationship between optimal solutions of primal (P) and dual (D) problems. Under certain convexity assumptions and constraint qualification conditions, the optimal primal and dual objective function values match at optimality (for details, refer [5]). This scenario (that is, when $f^P = \theta^D$) is referred to as *zero duality gap* scenario. The *duality gap* (ρ) is the difference between the objective value of the optimal primal solution and that of the optimal dual solution, or $\rho = (f^P - \theta^D)$. The following results can be obtained from the mathematical optimization literature.

1. For any optimization problem, if $\mathbf{x}$ and $\boldsymbol{\lambda}$ are any *feasible* solutions to problems P and D, respectively, then $f(\mathbf{x}) \geq \theta(\boldsymbol{\lambda})$. This principle is called the *weak duality theorem* [5, 32].

2. Furthermore, for any primal problem, the optimal primal objective value is always greater than or equal to the optimal dual objective value. In other words, $f^P \geq \theta^D$ or, $\rho \geq 0$.

3. For convex primal problems, the duality gap is zero, thereby establishing $f^P = \theta^D$ or, $\rho = 0$.

4. The point $(\mathbf{x}^P, \boldsymbol{\lambda}^D)$ is a *saddle* point of the Lagrangian function, that is, the condition $L(\mathbf{x}^P, \boldsymbol{\lambda}) \leq L(\mathbf{x}^P, \boldsymbol{\lambda}^D) \leq L(\mathbf{x}, \boldsymbol{\lambda}^D)$ is true for all feasible $\mathbf{x}$ and $\boldsymbol{\lambda}$.

Although for all convex problems, the duality gap must be zero, this does not mean that for every non-convex optimization problem, the duality gap will be strictly greater than zero. Certain non-convex problems may also exhibit zero duality gap.

The above results motivate researchers to look for the solution of the Lagrangian dual problem (D) for several reasons:

1. The dual problem (D) is always concave, meaning that the negative of the Lagrangian dual function $\theta(\boldsymbol{\lambda})$ is always a convex function. Thus, if the Lagrangian dual subproblem can be solved exactly, the dual problem is comparatively easier to solve, although the original primal problem, P, may be harder to optimize.

2. If in a problem, the number of decision variables n is much greater than the number of constraints, m, that is, $n \gg m$, the dual formulation helps in changing the role of the decision variables and constraints. Thus, the dual problem has a smaller dimensional search space, although the computation of the objective function requires solution of another optimization problem.

3. Due to the fact that duality gap is always non-negative, the optimal dual function value provides a *lower bound* of the optimal primal function value. That is, if the dual problem can be solved, its optimal objective value will provide a lower bound to the optimal objective value of the primal problem.

4. For problems having a zero duality gap, if the optimal dual solution ($\boldsymbol{\lambda}^D$) exists, it is identical to the Lagrange multiplier vector corresponding to the optimal primal problem. The Lagrange multipliers are useful in performing a *sensitivity* analysis to the original optimization problem and are useful from a practical standpoint [9, 31, 30].

If viewed closely, the dual problem is also a max-min problem (if the Lagrangian subproblem has a bounded solution). Min-max and max-min problems commonly appear in many scientific and engineering problem-solving tasks. Thus, an efficient method of solving the Lagrangian dual problem is also expected to be useful for solving min-max or max-min problems.

The above discussions amply indicate the importance of solving the Lagrangian dual problem efficiently. Due to the inherent link between primal variables ($\mathbf{x}$) and dual variables ($\boldsymbol{\lambda}$), in this paper, we propose a *coevolutionary optimization algorithm* in which evolution of a population of primal variables is intimately tied with the evolution of a population of dual variables. In the next section, we review some of the existing algorithms for solving the dual problem and thereafter present our proposed coevolutionary approach.

3 Existing Solution Methodologies for the Dual Problem

There exists a number of strategies by which researchers have approached solving a dual problem in the past. However, in most such cases, the study is confined to solving linear programs, quadratic programs, or, in general, convex programming problems. Since the optimal primal solution is directly linked to the optimal dual solution for convex problems, such a study has helped researchers understand the link better.

Goldfarb and Idnani [20] suggested a successive quadratic programming based algorithm to solve convex quadratic programs and demonstrated that the dual method is computationally superior compared to the direct QP algorithm in solving the primal problem, particularly when a feasible solution to the primal problem is difficult to achieve. A cutting plane method was then proposed by Bazaraa et al. [5], where the dual problem is converted into a linear program and the solution is searched through an iterative scheme. Malek et al. [25] has solved the primal-dual problem for the linear programming problems only.

One of the difficulties in optimizing the dual problem is that the Lagrangian dual sub-function ($\theta(\boldsymbol{\lambda})$) is often non-differentiable at its maximum point (exhibiting a 'kink'). Thus, the usual gradient-based optimization algorithms employed to solve the dual subproblem usually fails to capture the exact maximum point. In such cases, the use of a *subgradient* is a viable method in solving the dual problem [32, 27]. For this purpose, the bundle method [26, 36], in which several subgradients are computed at nearby points and a quadratic programming problem is solved to find the best convex combination of them, is a common approach. Among the other methods, the space dialation methods and subgradient-projection methods are suggested [29, 33]. Due to the non-differentiability of the θ function at its optimum, meta-heuristic approaches like evolutionary algorithms (EAs) stand as suitable choices for optimizing the dual problem.

Although not directly used to solve dual problems, a coevolutionary genetic algorithm (CGA) was suggested by Barbosa [2] and later by [22] to solve min-max problems of the following type:

$$\min_{\mathbf{x}\in X}\max_{\mathbf{y}\in Y}\ U(\mathbf{x},\mathbf{y}). \tag{4}$$

In the above problem, for a particular solution $\bar{\mathbf{x}}$, the inner-level (or lower-level) subproblem of maximizing $U(\bar{\mathbf{x}},\mathbf{y})$ with respect to $\mathbf{y}$ must have to done. This will result in an optimal solution $\mathbf{y}^*(\bar{\mathbf{x}})$ which is a function of $\bar{\mathbf{x}}$. Thereafter, in the outer-level (or upper-level) optimization the function $U(\mathbf{x},\mathbf{y}^*(\mathbf{x}))$ has to be minimized with respect to $\mathbf{x}$ to find the min-max solution of the problem. It is trivial to note that inherently a min-max problem and a dual problem are similar to each other, except in the dual problem the lower-level problem is a minimization problem and the upper level problem is a maximization problem. In the coevolutionary algorithm suggested by Barbosa [2], two populations A and B are used to represent the two sets of variables – $\mathbf{x}$ and $\mathbf{y}$. The fitness of a member $\mathbf{x}^i$ in population A and $\mathbf{y}^j$ in population B were calculated as:

$$\begin{aligned} \mathcal{F}_{\mathbf{x}^i} &= \max_{\mathbf{y}\in B} U(\mathbf{x}^i,\mathbf{y}),\\ \mathcal{F}_{\mathbf{y}^j} &= \min_{\mathbf{x}\in A} U(\mathbf{x},\mathbf{y}^j). \end{aligned} \tag{5}$$

Thus, the fitness of one population largely depends on the other population in a truly coevolutionary sense. These fitness values are then used to evolve both populations, A and B for a fixed number of generations each. The population A is evolved for a fixed *max-gen-A* number of generations and the population B is evolved for a fixed *max-gen-B* number of generations. By evolving the populations in tandem while keeping the other population unaltered, the study hoped to reach near the min-max solution of a number of two-variable problems.

However, it is already highlighted in another study [24] and it is important to note here that the above min-max problem (or, for this matter, dual problem) is not *symmetric* in terms of its upper and lower optimization tasks. The upper-level problem takes the role of a 'leader' and the lower-level task must act as a 'follower' to the upper-level problem in providing the optimal $\mathbf{x}$-vector to the corresponding lower-level subproblem. However, the above-mentioned coevolutionary approach emphasizes both problems 'equally' or 'symmetrically' for it to be an efficient strategy for solving the min-max or the dual problem. Jensen [24] criticized the use of CGA approach in solving generic min-max problems and clearly mentioned that they will work well in problems for which the solution of the min-max problem is identical to that of the max-min problem obtained by exchanging the role of the two subproblems. Although not argued in the context of primal-dual optimization problems, Jensen's argument can be extended to mean that the CGA approach will work for problems having a zero duality gap and may not work well on problems having non-zero duality gap.

To make coevolutionary principles to be effective for solving min-max problems, Jensen suggested a new *asymmetric* fitness evaluation scheme in which although the upper-level population is evaluated identically as that in CGA, the lower-level population is evaluated differently. First, for every upper-level population member, the lower-level population members are assigned a sorted rank from worst (highest) to best (smallest) with respect to $U(\mathbf{x}, \mathbf{y})$. Thereafter, the maximum rank obtained for all upper-level population members is assigned as a fitness to a lower-level population member. To differentiate two members having the same maximum rank, the member having higher frequency of the maximum rank is emphasized. This asymmetric coevolutionary genetic algorithm (ACGA) is found to produce better performance on a number of two-variable min-max problems in Jensen's study [24]. Since the dual problem is a max-min problem, the concept of ACGA can also be used to solve dual problems. We implement the ACGA approach for this purpose and compare it with our proposed methodology on standard constrained test problems in Section 6.

Apart from the asymmetric nature of min-max or dual problems, there is another major difficulty which has not been discussed much in the existing literature in solving such problems using a coevolutionary approach. The coevolutionary approaches of evolving two populations having an asymmetric importance, the accuracy of evaluating lower-level population members is of vital importance to the success of the overall procedure. Since in a coevolutionary approach lower-level population members are not evaluated to their optimality, the process introduces a *noise* in evaluating the upper-level population members. For a high-dimensional or a complex problem, such evaluation noise may not lead the approach to progress towards the true optimal solution. We discuss this issue further in Section 4.2.

Following Jensen's study, Branke and Rosenbusch [8] suggested four modifications to assign fitness to the lower-level population members differently. Although the study concentrated on handling worst-case scenario in an uncertainty-based optimization task using a coevolutionary approach, the concepts are interesting and can probably be used with some modifications for solving dual problems. Due to the indirect relationships between worst case optimization and dual problem solving task, we belabor these implementations for a later study. One of the criticisms of the existing coevolutionary studies including Branke and Rosenbusch's method [8] and Jensen's method [24] is that the proposed methodologies were used to solve simplistic and low-dimensional problems. With a few variables in both levels, most algorithms may demonstrate their successes, but the real test of an algorithm occurs when it is tried on relatively large-dimensional and complex problems.

Next, we would like to highlight that the dual problem can be considered as a special optimistic *bilevel* optimization problem [13, 14, 15, 16, 17]. A typical bilevel optimization problem (sometimes also known as a *Stackelberg game*), having two sets of variables $\mathbf{x}$ and $\mathbf{y}$ with sizes n and m, respectively, can be written as follows:

$$\begin{array}{rl} \text{minimize}_{(\mathbf{x},\mathbf{y})} & F(\mathbf{x},\mathbf{y}), \\ \text{subject to} & \mathbf{x} \in \text{argmin}_{\mathbf{z}} \left\{ f(\mathbf{z},\mathbf{y}) | \mathbf{z} \in X \right\}, \\ & (\mathbf{x} \times \mathbf{y}) \in (X \times Y). \end{array} \tag{6}$$

Here, the upper-level problem minimizes the objective $F : \Re^n \times \Re^m \rightarrow \Re$, subject to satisfying a set of constraints including one that involves a lower-level minimization problem. For the lower-level problem, the upper-level variable vector $\mathbf{y}$ is fixed and optimal solutions to the lower-level objective $f : \Re^n \rightarrow \Re$ are feasible candidates to the upper-level problem. Although the problem looks complex, bilevel problems commonly appear in practice [1, 4, 6, 18, 23]. A comparison between problems given in Equations 3 and 6 reveals the following:

$$\text{Dual problem} \qquad \qquad \text{Bilevel problem} \tag{7}$$
$$\mathbf{x} \quad \Longrightarrow \quad \mathbf{x} \tag{8}$$
$$\boldsymbol{\lambda} \quad \Longrightarrow \quad \mathbf{y} \tag{9}$$
$$-L(\mathbf{x}, \boldsymbol{\lambda}) \quad \Longrightarrow \quad F(\mathbf{x}, \mathbf{y}) \tag{10}$$
$$L(\mathbf{x}, \boldsymbol{\lambda}) \quad \Longrightarrow \quad f(\mathbf{x}, \mathbf{y}) \tag{11}$$
$$\mathbf{x} \in X \quad \Longrightarrow \quad \mathbf{x} \in X \tag{12}$$
$$\boldsymbol{\lambda} \geq 0 \quad \Longrightarrow \quad \mathbf{y} \in Y \tag{13}$$

The specialty comes in the fact that both the upper and lower-level objectives are identical except in the sign. In the lower-level optimization problem, the variable set $\boldsymbol{\lambda}$ is fixed. In the upper-level optimization problem, $\mathbf{x}$ is fixed to one of the optimal solutions to the lower-level problem. Moreover, in the dual problem, constraints to the upper-level problem constitutes the non-negativity of the $\boldsymbol{\lambda}$ variables alone, that is, $\boldsymbol{\lambda} \geq 0$. Since the dual problem is a special-class of a generic bilevel optimization problem (in which F and f can be different), some of the existing bilevel EAs [21, 28, 35], although can be applied but may be too generic, to solve the Lagrangian dual problem discussed above. In the following section, we discuss two nested approaches that can be used to specifically solve dual problems and then present our proposed coevolutionary dual optimization algorithm.

3.1 Nested Algorithms

The dual problem can be solved in a nested manner by considering the following strategy. For every $\boldsymbol{\lambda}$-vector (a solution to the upper-level problem), the lower-level problem is solved *exactly* or with a numerical optimization algorithm run to achieve a desired precision in the optimized solution. The precise solution of Lagrangian dual subproblem will be a key operation in finding the dual solution from the upper-level problem. However, this can be computationally quite expensive, as for every upper-level solution ($\boldsymbol{\lambda}$), an optimization problem needs to be solved with a reasonable precision. To compare, we consider two methodologies for this purpose. In the approaches, for the lower-level optimization task, we employ a classical point-by-point optimization algorithm, however for the upper-level optimization task we can use either a classical method or an evolutionary algorithm.

In the first 'classically nested' or N-Cl approach, both lower and upper-level optimization tasks are solved using Matlab's `fmincon()` routine. In both levels, there is only one solution at each iteration. For a $\bar{\boldsymbol{\lambda}}$-vector, the lower-level optimization problem is solved by `fmincon()` routine using a randomly chosen $\mathbf{x}$-vector as initial point and the corresponding optimal $\bar{\mathbf{x}}$-vector is obtained. The $\bar{\mathbf{x}}$-vector is then combined with $\bar{\boldsymbol{\lambda}}$-vector and used to compute the upper-level function value $\theta(\bar{\boldsymbol{\lambda}})$. The upper-level optimization task is solved by another meta-level `fmincon()` routine for which $\boldsymbol{\lambda}$-vector is the set of variables. At the end of the nested optimization task, the best $\boldsymbol{\lambda}^b$ solution and its corresponding optimal $\mathbf{x}^b$-vector are reported. Since the combination $(\boldsymbol{\lambda}^b, \mathbf{x}^b)$ provides a solution to the upper-level problem, the overall number of solution evaluations needed in both optimization tasks are recorded and considered as a performance indicator to the N-Cl procedure.

Since in most problems the upper-level objective function ($\theta(\boldsymbol{\lambda})$) is usually non-differentiable at its optimum, classical gradient-based methods may not work very well in such problems and a subgradient approach may be used. However, since GAs can be used to solve non-differentiable problems, as a second approach, here we employ a real-parameter genetic algorithm (RGA) to solve the upper-level optimization problem, but the lower-level problem is still solved using the `fmincon()` routine. We call this method the 'hybrid nested' or N-GA approach. The RGA approach uses the simulated binary recombination (SBX) operator [11] and the polynomial mutation operator [10] with their standard parameter values. The RGA procedure uses a generational GA which uses the binary tournament selection operator without replacement and variable-wise blending of parent decision variables to create new real values for offspring population members. No specific elite preserving mechanism is used here, however, the population-best $\boldsymbol{\lambda}$ vector and its corresponding optimal $\mathbf{x}$ vector (`fmincon()` solution obtained from $\boldsymbol{\lambda}$) are preserved from one generation to the next. The initial population members ($\mathbf{x}$-vector for lower-level population and $\boldsymbol{\lambda}$-vector for upper-level populations) are created at random. Except in the very first generation, in which the corresponding $\mathbf{x}$-vector is used as a starting point for the `fmincon()` routine, in subsequent generations, the $\mathbf{x}$-v which is the partner of the best $\boldsymbol{\lambda}$-vector at the previous generation is used as the starting solution for the lower-level optimization task. To compare this method with other methods, the overall number of solution evaluations needed in both optimization algorithms is recorded.

4 Proposed Coevolutionary Dual Optimization Algorithm

In this section, we propose a coevolutionary dual optimization algorithm (CEDO) in which instead of one there are two populations that evolve in an intertwined manner. Each GA uses the usual tournament selection, SBX recombination, and polynomial mutation operators, but the computation of *fitness* value for each population member in both populations is revised to respect the meaning of dual problems. The dual problem presented in Equation 3 consists of two kinds of decision vectors: $\mathbf{x}$, the primal variables that are variables in the lower-level problem and $\boldsymbol{\lambda}$, the Lagrange multipliers that are variables in the upper-level problem. The formulation requires finding the infimum of the Lagrange function $L(\mathbf{x}, \boldsymbol{\lambda})$ with respect to $\mathbf{x}$ and then finding the maximum of the Lagrangian dual function $\theta(\boldsymbol{\lambda})$ with respect to $\boldsymbol{\lambda}$. We use a population $P_{\boldsymbol\lambda}$ of size $N_{\boldsymbol\lambda}$ to represent $\boldsymbol\lambda$ variables and population $P_{\mathbf{x}}$ of size $N_{\mathbf{x}}$ to represent $\mathbf{x}$ variables. We now discuss the fitness computation and population update procedures for each population.

4.1 Fitness Assignment and Population Update Procedures

For the upper-level problem, the fitness (for maximization) of i-th population member $\boldsymbol\lambda^i$ is *ideally* defined as $\mathcal{F}_{\boldsymbol\lambda}(\boldsymbol\lambda^i) = L(\mathbf{x}^{\boldsymbol\lambda^i,*}, \boldsymbol\lambda^i) = \theta(\boldsymbol\lambda^i)$, where $\mathbf{x}^{\boldsymbol\lambda^i,*}$ is the $\mathbf{x}$-vector corresponding to the infimum of $L(\mathbf{x}, \boldsymbol\lambda^i)$ for a fixed $\boldsymbol\lambda^i$-vector. However, instead of computing the exact optimum vector $\mathbf{x}^{\boldsymbol\lambda^i,*}$, we propose to use the best $\mathbf{x}$ member from the $P_{\mathbf{x}}$ population for $\boldsymbol\lambda^i$. Thus, the fitness of population member $\boldsymbol\lambda^i$ is defined as follows:

$$\mathcal{F}_{\boldsymbol\lambda}(\boldsymbol\lambda^i) = \max_{\mathbf{x}^j \in P_{\mathbf{x}}} L(\mathbf{x}^j, \boldsymbol\lambda^i). \tag{14}$$

The corresponding population member (say, $\mathbf{x}^{\boldsymbol\lambda^i}$) in $P_{\mathbf{x}}$ that maximizes the above $L()$ function becomes a *partner* of the corresponding $\boldsymbol\lambda^i$-vector. The above operation requires that for every $P_{\boldsymbol\lambda}$ population member, all $P_{\mathbf{x}}$ population members have to be checked one by one and the one that will correspond to the maximum value of the Lagrange function will be chosen as its partner. It is important to note that this process may make a single $P_{\mathbf{x}}$ member as a partner for more than one $P_{\boldsymbol\lambda}$ members. The $P_{\boldsymbol\lambda}$ population members can then be selected in the tournament selection based on the above objective value.

As indicated earlier and is also discussed in an earlier study [24] in the context of a min-max problem, the dual problem, being a specific class of a generic bilevel programming problem, is not symmetric in its importance to upper and lower-level subproblems. The dual problem must emphasize the lower-level members that are partners of good upper-level members in order to steer the search towards the dual solution. Thus, we do not perform an independent selection operation on population $P_{\mathbf{x}}$. Instead, after a $\boldsymbol\lambda$ member is chosen by the tournament selection operator, its corresponding $\mathbf{x}$ partner is also automatically selected from $P_{\mathbf{x}}$ and placed into its own mating pool. This way, a linkage between $\boldsymbol\lambda$-$\mathbf{x}$ vectors is established and the pair is emphasized in both populations. The above procedure provides a selective pressure for a lower-level population member that is closely related to how its partner fairs on the upper-level population. If a $\mathbf{x}$ member happens to be so good that it is a partner with more than one $\boldsymbol\lambda$ members, then its chance of surviving by the above selection operator is high. Recombination and mutation operators are then independently applied to both populations using the usual SBX and polynomial mutation operators [10, 11].

4.2 Noisy Evaluation and Use of a Local Search

A little thought will indicate that the above fitness assignment scheme for a $P_{\boldsymbol\lambda}$ member is entirely dependent on $P_{\mathbf{x}}$ members and selection of $P_{\mathbf{x}}$ member depends on its partner in $P_{\boldsymbol\lambda}$ population, thereby constituting a coevolutionary effect to the overall algorithm. However, it is clear that the fitness $(L(\mathbf{x}^{\boldsymbol\lambda^i}, \boldsymbol\lambda^i))$ assigned to a $P_{\boldsymbol\lambda}$ member $(\boldsymbol\lambda^i)$ may be far from its ideal dual function value $(L(\mathbf{x}^{\boldsymbol\lambda^i,*}, \boldsymbol\lambda^i) = \theta(\boldsymbol\lambda^i))$. The noise introduced in the fitness computation for the i-th upper-level population member $(\boldsymbol\lambda^i)$ is given as follows:

$$\varepsilon(\boldsymbol\lambda^i) = L(\mathbf{x}^{\boldsymbol\lambda^i}, \boldsymbol\lambda^i) - L(\mathbf{x}^{\boldsymbol\lambda^i,*}, \boldsymbol\lambda^i). \tag{15}$$

Since the lower-level problem is a minimization problem, the first term is always larger than the second term and the above error term is always non-negative. The fitness of an individual in $P_{\boldsymbol\lambda}$ is noisy due to this error and the noise makes the task of emphasizing the correct solution difficult. To illustrate, in Figure 1, we show the fitness distribution of a $P_{\boldsymbol\lambda}$ population for a two-variable, single-constraint problem:

$$\begin{array}{ll} \text{minimize} & f(x_1, x_2) = (x_1 - 1)^2 + (x_2 - 1)^2, \\ \text{subject to} & g(x_1, x_2) \equiv x_1 + x_2 \leq 1. \end{array} \tag{16}$$

Both $P_{\boldsymbol\lambda}$ and $P_{\mathbf{x}}$ population members are shown in Table 1. The ideal dual function value for each λ value is shown in the last column of Table 1. However, the fitness value $(L(\mathbf{x}^\lambda, \lambda))$ of each $P_{\boldsymbol\lambda}$ population member computed using the $P_{\mathbf{x}}$ population is shown in column 6. The figure also clearly shows this difference. The parabola shown in the figure represents the ideal dual function value $(L(\mathbf{x}^{\lambda,*}, \lambda))$ as a function of λ, however, for the given $P_{\mathbf{x}}$ population, the fitness value of each $P_{\boldsymbol\lambda}$ population member is shown by open circles. The fitness value indicates that solution A is the winner (having highest $L(\mathbf{x}^\lambda, \lambda)$ value), although ideally solution B must be the true winner having a higher $L(\mathbf{x}^{\lambda,*}, \lambda)$ or $\theta(\lambda)$ value. Thus, if solutions A

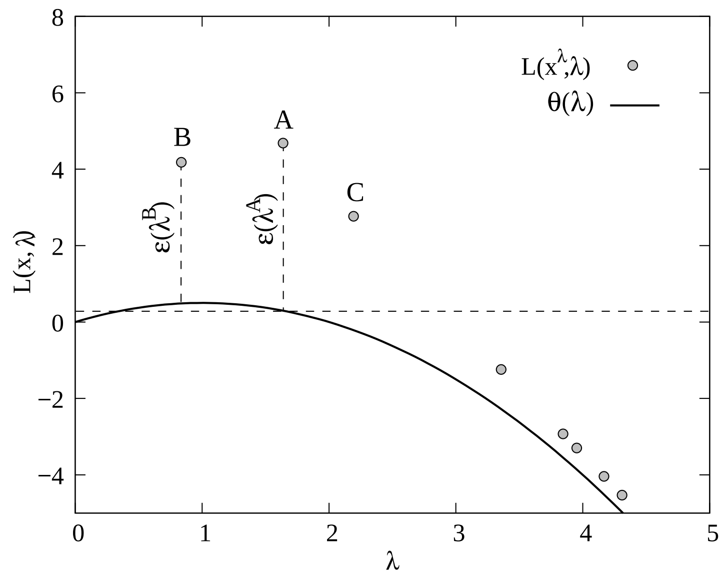


Figure 1: A sketch illustrating the selection of wrong individual due to the presence of error mentioned in Equation 15.

Table 1: Upper and lower-level population members ($N_{\boldsymbol{\lambda}} = 10$ and $N_{\mathbf{x}} = 10$) for the problem given in equation 16.

Sol.	$\boldsymbol{\lambda}$	x_1	x_2	$L(\mathbf{x}^{\lambda}, \lambda)$	$L(\mathbf{x}^{\lambda,*}, \lambda)$
	3.952	2.363	−1.053	−3.297	−3.857
	4.747	1.834	2.040	−6.037	−6.518
A	1.638	−0.577	−4.804	4.683	0.297
	3.356	−1.691	−0.757	−1.243	−2.276
C	2.193	−2.297	−3.029	2.768	−0.212
	4.168	3.217	−0.701	−4.040	−4.517
	3.844	3.878	−1.088	−2.926	−3.545
B	0.836	2.691	−1.032	4.182	0.487
	4.310	3.085	2.551	−4.531	−4.978
	4.949	−1.226	−2.840	−6.737	−7.299

and B are compared in the tournament selection of our proposed CEDO algorithm, solution A will be chosen, because its apparent fitness value is larger than that of B. The fitness of an individual is the only source of information that is exploited by the operators of an evolutionary algorithm in order to produce better offspring as the generation proceeds. If such errors are consistently made by the proposed algorithm, it is likely that it will proceed in the wrong direction and may not eventually converge to the true dual solution.

In order to alleviate the noise associated with our fitness assignment, we take help of a local search operation that will optimize the lower-level problem using a classical optimization tool, but importantly use this local search operation *sparingly* as and where needed. It is clear that if both solutions A and B were local searched, their fitness values will get modified to reach the parabolic function shown in the figure. Now, a tournament selection operation with these exact fitness values will select the right candidate (B). But there is a computational difficulty, which we wanted to negotiate right from the beginning. It is intuitive to note that if the local search operation is performed for *all* $P_{\boldsymbol{\lambda}}$ population members, this approach is nothing but the hybrid nested (N-GA) approach discussed earlier. To constitute a computationally tractable algorithm, we perform the local search to only the best $P_{\boldsymbol{\lambda}}$ population member ($\boldsymbol{\lambda}^b$) in every generation. Here, we use `fmincon()` routine for this purpose. Although early on, this is a good idea, in later generations when the population members are close to the dual solution, the population members are likely to be close to each other and performing local search even for a single best $\boldsymbol{\lambda}$ population member for all generations from start to finish may be too expensive. To reduce the computational effort, we use an *archive* of local-searched solutions right from the initial generation. Any solution $\boldsymbol{\lambda}^b$ that is local-searched is stored in the archive with its corresponding $\mathbf{x}^{\boldsymbol{\lambda}^b}$ vector. When a new solution $\boldsymbol{\lambda}$ is to be local-searched, first the archive is searched to locate if there is any archive member that is close to the new solution. We use a normalized Euclidean distance measure for this purpose. If the distance value is smaller than a pre-defined threshold d_{allow}, the new solution is considered to be too similar to an archive member that has been local searched already, and is not local-searched further. Its partner $\mathbf{x}$-vector and fitness value $\mathcal{F}_{\boldsymbol{\lambda}}$ are assigned as the corresponding values. However, if the new solution is not close to any archive member, it is modified using the local search procedure.

The performance and required computational effort of a local search operation largely depends on the chosen initial starting solution. We reduce the effort by using the partner $\mathbf{x}$-vector of $\boldsymbol{\lambda}^b$ as the starting solution to the local search procedure. To further reduce the computational effort, we run each local search procedure for a small pre-specified number (T_{local}) of iterations.

The main objective of using the local search is to reduce the noise in evaluating the fitness of the population-best $P_{\boldsymbol{\lambda}}$ member. We aim to reduce the noise to the extent that the current best member in the population $P_{\boldsymbol{\lambda}}$ at any generation is free from noise. Performing the local search on the best individual $\boldsymbol{\lambda}^b$ in a generation can lead to two possible scenarios. The lower-level function value ($L()$) for $\boldsymbol{\lambda}^b$ has either reduced or remained the same. If the fitness remained the same, it implies that its partner in $P_{\mathbf{x}}$ population was in fact the theoretical infimum solution to the corresponding $L()$ function. Thus, the best solution remains still the best in the population. However, if the fitness has reduced after the local search operation, there may be two scenarios. If the decrease is to such an extent that the ideal $L()$ value for $\boldsymbol{\lambda}^b$ is higher than the next-best fitness originally computed for the $P_{\boldsymbol{\lambda}}$ population members, it implies that the noise introduced by our evaluation did not affect the selection of the other population members. On the other hand, if the decrease in the fitness is such that the ideal $L()$ for $\boldsymbol{\lambda}^b$ is smaller than the fitness of a set of current $\boldsymbol{\lambda}$ population members, then declaring $\boldsymbol{\lambda}^b$ member as the best population member is doubtful. In such a scenario, we take each and every $\boldsymbol{\lambda}$ population member whose fitness is higher than the ideal $L()$ value of the best $\boldsymbol{\lambda}^b$ solution and perform a local search to these solutions one at a time.

In the context to Figure 1, a local search on solution A will decrease its $L()$ value and reach to the parabola, thereby making this fitness value worse than the fitness values of B and C. Thus, according to the above argument, A cannot be judged as the population-best solution and we should also apply local search to solutions B and C to check if one of them is, in fact, a better λ solution. As soon as a situation occurs when the current best member is still found to be best after local search operations of potential good solutions in $P_{\boldsymbol{\lambda}}$, no further local search operations will be made. For the case illustrated in Figure 1, after the local search of B, solution C must also be local searched. However, after solution C is local-searched and its $L()$ value reaches the parabola, no other population member has a higher fitness than best fitness of local-searched solutions A, B, and C. Thus, the local searching process will terminate after three local searches, despite having 10 members in $P_{\boldsymbol{\lambda}}$. In every case of local search, the archive is checked for the existence of a near-$\boldsymbol{\lambda}$ solution, and, if not, it and its new local-searched partner $\mathbf{x}$ are both entered in the archive.

The overall coevolutionary dual optimization algorithm is terminated when the mean fitness of the best $P_{\boldsymbol{\lambda}}$ population member in the past τ generations has not changed significantly. For this purpose, the standard deviation of $\mathcal{F}_{\boldsymbol{\lambda}}$ of the best $P_{\boldsymbol{\lambda}}$ population members in the past τ generations is computed and if it is less than a user-defined parameter δ, the algorithm is terminated and the current best $\boldsymbol{\lambda}$ solution and its partner $\mathbf{x}$ are reported. Total solution evaluations in CEDO and in local search operations are counted and reported for comparing with other nested methods discussed earlier.

4.3 Recommended Parameter Settings

The proposed CEDO algorithm has a few parameters to be set. After performing a number of runs on a number of test problems, we have found that there exist some values for which the proposed CEDO algorithms works well on most problems. They are suggested here:

Population sizes: The upper and lower population sizes are set as follows:

$$N_{\lambda} = N_{\mathbf{x}} = 10\min(n, m),$$

where n is the number of decision variables and m is the number of constraints. This is in tune with existing GA studies in which the population size is set in proportion to the number of variables [19]. In all our simulations, we have kept both population sizes the same, but it is clearly not a mandatory requirement.

Maximum `fmincon()` iterations: The parameter T_{local} is set to 10, as this value was found to provide a good trade-off between accuracy and computational effort.

Threshold for similarity of best population member with an archive member: This measure is used to determine if an archive member close to $\boldsymbol{\lambda}^b$ exists, so that it need not be modified unnecessarily with a local search. A value of $d_{\text{allow}} = 0.05$ is found to work well on all problems of this study.

Termination condition and parameters: At every generation after τ generations, the average function value of the population-best solution for the past τ generations is compared with the function value of the known dual solution. If the absolute difference is less than a parameter δ, the algorithm is terminated. Otherwise, the algorithm terminates at the maximum specified number of generations, which is taken as 2,000 and in this case, the run is declared unsuccessful. We have used $\tau = 20$ and $\delta = 0.0005$ for all problems here. For an arbitrary problem in which target dual solution may not be known a priori, the standard deviation in population-best solution for the past τ generation can be checked with δ to determine the termination of a run.

Evolutionary algorithm parameters:

Probability of SBX recombination, $p_c^{\boldsymbol{\lambda}} = p_c^{\mathbf{x}}$	$= 0.9$
Probability of mutation, $p_m^{\boldsymbol{\lambda}}$	$= 1/m$,
Probability of mutation, $p_m^{\mathbf{x}}$	$= 1/n$,
Distribution indices η_c, η_m	$= 2$, 100
Lower and Upper bound on $\mathbf{x}$	$= -10000$, 10000
Lower and Upper bound on $\boldsymbol{\lambda}$	$= 0$, 10000

The lower and upper bound of $\mathbf{x}$ and $\boldsymbol{\lambda}$ variables mentioned above are used as fixed bounds in performing the crossover and mutation operators. However for initialization purpose, we have used smaller bounds: $\mathbf{x}_i \in (-20, 20)$ and $\boldsymbol{\lambda}_i \in (0, 20)$. In problems having infinite duality gap, some variable values may be unbounded and keeping a large range of values for $\mathbf{x}$ variables helps to get an indication of this effect. These bounds have been used consistently in all the test problems considered in this paper. It is important to highlight that above allowable ranges in $\mathbf{x}$ and $\boldsymbol{\lambda}$, although provides a flexibility in the search to locate the correct dual solution, the flexibility comes with a price of possible premature convergence particularly in solving complex problems and using algorithms that are not adequately efficient. But we have chosen these wide ranges in $\mathbf{x}$ and $\boldsymbol{\lambda}$ variables to provide the algorithm with a stringent test and also facilitate solving infinite duality gap problems, in which theoretical values of some variables may take positive or negative infinity.

For two existing evolutionary approaches (CGA and ACGA), identical recombination and mutation operators and parameters to that used for CEDO are used. Population size and termination condition are also kept identical to make a fair comparison. For the N-Cl and N-GA, parameters and termination conditions are also kept the same.

To show the robustness of the proposed algorithm, each problem is solved 25 times with different initial populations. Best, median and worst performance of 25 runs are then presented to get an idea of the sensitivity of the proposed algorithm on chosen initial populations.

5 Test Problems

Here, we present 11 test problems used to demonstrate the working of our proposed CEDO algorithm. Note that each of the variable bounds is considered as an additional inequality constraint. Thus, the set X (in Equation 1) is considered to be the complete n-dimensional real space, or $X = \Re^n$.

5.1 Problem P1

It is a two-variable problem having two constraints:

$$\begin{array}{ll} \text{minimize} & (x_1-2)^2+(x_2-2)^2, \\ \text{subject to} & x_1+x_2-3 \le 0, \\ & x_2-10x_1+2 \le 0. \end{array} \tag{17}$$

The optimal primal solution $\mathbf{x}^P = (1.5, 1.5)$ with $f(\mathbf{x}^P) = 0.5$ and its corresponding Lagrange multiplier vector $\boldsymbol{\lambda}^P = (1, 0)$ can be obtained by analyzing the KKT optimality conditions. The $\boldsymbol{\lambda}^P$ indicates that only the first constraint is active at $\mathbf{x}^P$. This problem is convex and hence that this problem has a zero duality gap.

5.2 Problem P2

This problem is studied elsewhere [28] and has $n = 2$ and $m = 2$:

$$\begin{array}{ll} \text{minimize} & (x_1-2)^2+(x_2-1)^2, \\ \text{subject to} & x_1+x_2-2 \le 0, \\ & x_1^2-x_2 \le 0. \end{array} \tag{18}$$

This problem also has a convex feasible region and convex objective function. The primal solution corresponds to $\mathbf{x}^P = (1, 1)$, $f(\mathbf{x}^P) = 1$ with $\boldsymbol{\lambda}^P = (2/3, 2/3)$. Both constraints are active at $\mathbf{x}^P$.

5.3 Problem P3

This problem is non-convex and corresponds to $n = 2$ and $m = 6$ (including variable bounds):

$$\begin{array}{ll} \text{minimize} & 100(x_2-x_1^2)^2+(1-x_1)^2, \\ \text{subject to} & -x_1^2-x_2 \le 0, \\ & -x_1-x_2^2 \le 0, \\ & -0.5 \le x_1 \le 0.5, \quad 0 \le x_2 \le 1. \end{array} \tag{19}$$

The primal solution is given as $\mathbf{x}^P = (0.5, 0.25)$ having $f(\mathbf{x}^P) = 0.25$. The corresponding Lagrange multiplier vector is $\boldsymbol{\lambda}^P = (0, 0, 0, 1, 0, 0)$. Only $x_1 \le 0.5$ constraint is active at the optimum.

5.4 Problem P4

This is a non-convex problem and has $n = 2$ and $m = 2$:

$$\begin{array}{ll} \text{minimize} & (x_1^2+x_2-11)^2+(x_1+x_2^2-7)^2, \\ \text{subject to} & (x_1-0.05)^2+(x_2-2.5)^2-4.84 \le 0, \\ & 4.84-x_1^2-(x_2-2.5)^2 \le 0. \end{array} \tag{20}$$

The optimal primal solution lies on the first constraint boundary and the solution is given as follows: $\mathbf{x}^P = (2.2468, 2.3819)$, $f(\mathbf{x}^P) = 13.591$ and $\boldsymbol{\lambda}^P = (6.8835, 0)$.

5.5 Problem P5

This is a commonly-used test problem in EA literature and has $n = 7$ and $m = 4$:

$$\begin{array}{ll} \text{minimize} & (x_1-10)^2+5(x_2-12)^2+x_3^4+3(x_4-11)^2+10x_5^6+7x_7^4, \\ & \quad -4x_6x_7-10x_6-8x_7, \\ \text{subject to} & 2x_1^2+3x_2^4+x_3+4x_4^4+5x_5-127 \le 0, \\ & 7x_1+3x_2+10x_3^2+x_4-x_5-282 \le 0, \\ & 23x_1+x_2^2+6x_6^2-8x_7-196 \le 0, \\ & -4x_1^2+x_2^2-3x_1x_2+2x_3^2+5x_6-11x_7 \le 0. \end{array} \tag{21}$$

The first and fourth constraints are active at $\mathbf{x}^P$. The solution is reported elsewhere [12] and is given as follows: $\mathbf{x}^P$ =(2.3305, 1.9514, -0.4775, 4.3657, -0.6245, 1.0381, 1.5942), $f(\mathbf{x}^P) = 680.63$, and $\boldsymbol{\lambda}^P = (1.1397, 0, 0, 0.3686)$. The first and fourth constraints are active at this optimum.

5.6 Problem P6

This is another test problem often used in constrained EA studies. It has $n = 10$ and $m = 8$:

$$
\begin{array}{ll}
\text{minimize} & x_1^2 + x_2^2 + x_1x_2 - 14x_1 - 16x_2 + (x_3-10)^2 + 4(x_4-5)^2 + (x_5-3)^2, \\
& \quad +2(x_6-1)^2 + 5x_7^2 + 7(x_8-11)^2 + 2(x_9-10)^2 + (x_{10}-7)^2 + 45, \\
\text{subject to} & 4x_1 + 5x_2 - 3x_7 + 9x_8 - 105 \le 0, \\
& 10x_1 - 8x_2 - 17x_7 + 2x_8 \le 0, \\
& -8x_1 + 2x_2 + 5x_9 - 2x_{10} - 12 \le 0, \\
& 3(x_1-2)^2 + 4(x_2-3)^2 + 2x_3^2 - 7x_4 - 120 \le 0, \\
& 5x_1^2 + 8x_2 + (x_3-6)^2 - 2x_4 - 40 \le 0, \\
& x_1^2 + 2(x_2-2)^2 - 2x_1x_2 + 14x_5 - 6x_6 \le 0, \\
& 0.5(x_1-8)^2 + 2(x_2-4)^2 + 3x_5^2 - x_6 - 30 \le 0, \\
& -3x_1 + 6x_2 + 12(x_9-8)^2 - 7x_{10} \le 0.
\end{array} \tag{22}
$$

As reported in the literature, all but the final two constraints are active at $\mathbf{x}^P$ for this problem. The solution is given as follows: $\mathbf{x}^P$ =(2.1720, 2.3637, 8.7739, 5.0960, 0.9907, 1.4306, 1.3216, 9.8287, 8.2801, 8.3759) and $f(\mathbf{x}^P) = 24.3062$. The corresponding Lagrange multiplier vector is $\boldsymbol{\lambda}^P$ = (1.7165, 0.4745, 1.3759, 0.0205, 0.3120, 0.2870, 0, 0).

5.7 Problem P7

This problem is only one-dimensional ($n = m = 1$), but exhibits a finite duality gap. The constraint is non-differentiable at $x = 0$:

$$
\begin{array}{ll}
\text{minimize} & x^2(x^2/4 - 1), \\
\text{subject to} & |x| - 0.5 \le 0.
\end{array} \tag{23}
$$

The optimal primal solution is $x^P = \pm 0.5$ and objective value is $f^P = -0.234375$. The corresponding Lagrange multiplier is $\lambda^P = 0.875$. Due to the non-differentiability of this problem, it may be difficult to solve using gradient-based optimization algorithms.

5.8 Problem P8

This problem has three primal and eight dual variables:

$$
\begin{array}{ll}
\text{minimize} & x_1, \\
\text{subject to} & x_2 + x_3 \le 120, \\
& 100x_1 + 100x_2 - x_1x_3 + 100 \le 0, \\
& 10 \le x_1 \le 10000, \quad 10 \le x_2 \le 1000, \quad 10 \le x_3 \le 1000.
\end{array} \tag{24}
$$

The primal solution to this problem can be worked out using KKT optimality conditions and is given as follows: $\mathbf{x}^P = (110, 10, 110)$ with $f(\mathbf{x}^P) = 110$. The corresponding $\boldsymbol{\lambda}^P$ =(11, 0.1, 0, 0, 21, 0, 0, 0).

5.9 Problem P9

The problem is presented below and is commonly used in the constrained EA studies [12, 34]. The problem has $n = 8$ and $m = 11$ variables, thereby making it a reasonably large problem for finding the dual solution.

$$
\begin{array}{ll}
\text{minimize} & x_1 + x_2 + x_3, \\
\text{subject to} & 1000(-1 + 0.0025(x_4 + x_6)) \le 0, \\
& 1000(-1 + 0.0025(x_5 + x_7 - x_4)) \le 0, \\
& 1000(-1 + 0.01(x_8 - x_5)) \le 0, \\
& 0.001(-x_1x_6 + 833.33252x_4 + 100x_1 - 83333.333) \le 0, \\
& 0.001(-x_2x_7 + 1250x_5 + x_2x_4 - 1250x_4) \le 0, \\
& 0.001(-x_3x_8 + x_3x_5 - 2500x_5 + 1250000) \le 0, \\
& 100 \le x_1 \le 10000, \quad 1000 \le x_2 \le 10000, \quad 1000 \le x_3 \le 10000, \\
& 10 \le x_4 \le 1000, \quad 10 \le x_5 \le 1000, \quad 10 \le x_6 \le 1000, \\
& 10 \le x_7 \le 1000, \quad 10 \le x_8 \le 1000.
\end{array} \tag{25}
$$

The primal solution to this problem is calculated from the KKT optimality conditions elsewhere [34] and is shown below: $\mathbf{x}^P$ =(579.3169, 1359.343, 5110.071, 182.0174, 295.5985, 217.9799, 286.4162, 395.5979), with $f^P = 7049.3309$. The corresponding Lagrange multiplier vector was computed in the above-mentioned study and is given as follows: $\boldsymbol{\lambda}^P$ =(1.9665, 5.2173, 5.1165, 8.5465, 9.5908, 10.0128, 0, …, 0) is a 22-dimensional vector.

5.10 Problem P10

The following problem is non-convex having two primal variables and six dual variables.

$$
\begin{aligned}
\text{minimize} \quad & (x_1-10)^3+(x_2-20)^3, \\
\text{subject to} \quad & 100-(x_1-5)^2-(x_2-5)^2 \le 0, \\
& (x_1-6)^2+(x_2-5)^2-82.81 \le 0, \\
& 13 \le x_1 \le 100, \\
& 0 \le x_2 \le 100.
\end{aligned} \tag{26}
$$

The primal solution is obtained by a KKT optimality analysis [34] and is $\mathbf{x}^P = (14.095, 0.8429)$ with $f^P = -6961.7511$. The corresponding Lagrange multiplier vector, computed elsewhere, is given as follows: $\boldsymbol{\lambda}^P$ =(1097.1109, 1229.5333, 0, 0, 0, 0). The first two constraints are active at $\mathbf{x}^P$.

5.11 Problem P11

This problem has five primal variables and 16 dual variables. This problem is also extensively used in the EA literature.

$$
\begin{aligned}
\text{minimize} \quad & 5.3578547x_3^2+0.8356891x_1x_5+37.293239x_1-40792.141, \\
\text{subject to} \quad & -(85.334407+0.0056858x_2x_5+0.0006262x_1x_4-0.0022053x_3x_5) \le 0, \\
& 85.334407+0.0056858x_2x_5+0.0006262x_1x_4-0.0022053x_3x_5 \le 92, \\
& -(80.51249+0.0071317x_2x_5+0.0029955x_1x_2+0.0021813x_3^{)} \le -90, \\
& 80.51249+0.0071317x_2x_5+0.0029955x_1x_2+0.0021813x_3^2 \le 110, \\
& -(9.300961+0.0047026x_3x_5+0.0012547x_1x_3+0.0019085x_3x_4) \le -20, \\
& 9.300961+0.0047026x_3x_5+0.0012547x_1x_3+0.0019085x_3x_4 \le 25, \\
& 78 \le x_1 \le 102, \quad 33 \le x_2 \le 45, \\
& 27 \le x_i \le 45, \quad \forall\, i \in \{3,4,5\}.
\end{aligned} \tag{27}
$$

The best-known primal solution for this problem is reported to be $\mathbf{x}^P$ = (78, 33, 29.995, 45, 36.776) with $f^P = -30665.5$. The corresponding Lagrange multipliers are $\boldsymbol{\lambda}^P$ =(0, 403.2517, 0, 0, 809.4188, 0, 48.9268, 0, 84.3203, 0, 0, 0, 0, 26.6393, 0, 0), meaning second and fifth constraints are active and variable bounds $x_1 \ge 78$, $x_2 \ge 33$, $x_4 \le 45$ are also active.

6 Results and Discussions

The coevolutionary dual optimization algorithm and two nested algorithms are applied to all 11 test problems illustrated in Section 5. Based on our results, we have observed that these problems are categorized into three groups:

1. Zero duality gap: Problems P1 to P6,
2. Finite duality gap: Problems P7 to P9, and
3. Infinite duality gap: Problems P10 and P11.

Although some of these problems had been regularly used in the evolutionary computation literature since the past two decades, they were never studied enough to investigate the inherent duality gap that they possess. We provide mathematical reasons stating why the second and third category problems have finite or infinite duality gaps. It is noteworthy that although many mathematical optimization books and literature discuss theories on duality gap, so far, to our knowledge, no known numerical optimization problem is suggested to exhibit a finite duality gap. In this paper, for the first time, we have constructed one such problem having a single variable only and have also unveiled finite duality property of a couple of other existing test problems.

6.1 Zero Duality Gap Problems

Based on our results, we observe that the first six test problems (P1 to P6) qualify to be in the category of zero duality gap. For all these problems, we have found that the dual solution obtained by our coevolutionary dual optimization (CEDO) algorithm is identical or almost identical to their respective primal solutions. The results are presented in Table 2. We now investigate the computational burden needed to obtain the dual solutions. Table 3 shows the best, median and worst Lagrange function value obtained by 25 different runs of our CEDO algorithm, the two nested algorithms (N-Cl and N-GA), CGA and ACGA, discussed earlier. The smallest, median and largest number of solution evaluations (SE) needed to achieve the presented dual solutions in 25 runs are also shown in the table for all algorithms. We now discuss certain details of the obtained results for the six zero-duality test problems one by one.

Table 2: Dual solutions obtained by CEDO (second row) are compared with the optimal primal solutions (first row) reported in Section 5 for problems P1 to P6.

		$\boldsymbol{\lambda}^P$ (Primal), $\boldsymbol{\lambda}^D$ (CEDO)	$\mathbf{x}^P$ (Primal), $\mathbf{x}^D$ (CEDO)	f^P (Primal) θ^D (CEDO)
P1	Primal	(1, 0)	(1.5, 1.5)	0.5
	CEDO	(0.9984, 0.0000)	(1.5008, 1.5008)	0.5000
P2	Primal	(0.6667, 0.6667)	(1, 1)	1
	CEDO	(0.6595, 0.6695)	(1.0005, 1.0050)	1.0000
P3	Primal	(0, 0, 0, 1, 0, 0)	(0.5, 0.25)	0.25
	CEDO	(0.0000, 0.0000, 0.0000, 0.9978, 0.0001, 0.0000)	(0.5040, 0.2541)	0.2500
P4	Primal	(6.8835, 0)	(2.2468, 2.3819)	13.5910
	CEDO	(6.8839, 0.0000)	(2.2467, 2.3819)	13.5908
P5	Primal	(1.1397, 0, 0, 0.3686)	(2.3305, 1.9514, −0.4775, 4.3657, −0.6245, 1.0381, 1.5942)	680.6300
	CEDO	(1.1397, 0.0000, 0.0000, 0.3660)	(2.3356, 1.9516, −0.4807, 4.3651, −0.6243, 1.0403, 1.5922)	680.6300
P6	Primal	(1.7165, 0.4745, 1.3759, 0.0205, 0.3120, 0.2870, 0, 0)	(2.1720, 2.3637, 8.7739, 5.0960, 0.9907, 1.4306, 1.3216, 9.8287, 8.2801, 8.3759)	24.3062
	CEDO	(1.7156, 0.4739, 1.3775, 0.0204, 0.3132, 0.2871, 0.0000, 0.0000)	(2.1718, 2.3596, 8.7744, 5.0960, 0.9902, 1.4306, 1.3204, 9.8294, 8.2781, 8.3777)	24.3061

6.1.1 Results on Problem P1

We take advantage of two-variable nature of this problem to demonstrate how our CEDO algorithm works on this problem. Figures 2 to 7 show both $P_{\mathbf{x}}$ and $P_{\boldsymbol{\lambda}}$ populations at generations 1, 7,and 20, respectively. The initial population for $P_{\boldsymbol{\lambda}}$ and $P_{\mathbf{x}}$ were created randomly within $[0, 20]$ and $[-20, 20]$, respectively. At generation 1, $P_{\boldsymbol{\lambda}}$ members are shown in Figure 2, but population members that are partners of $P_{\boldsymbol{\lambda}}$ are shown in Figure 3. The feasible variable space is shown in Figure 3 by drawing the two constraint functions. In all figures, the dual $\boldsymbol{\lambda}$ solution and corresponding $\mathbf{x}$ solution are marked using diamonds. Figure 3 shows the contour plot of the objective function $f(\mathbf{x})$. It is clear that the first constraint is active at the optimum. Since the second constraint is non-active, λ_2 must be zero. It is interesting how both populations approach their respective optimal locations with generations.

The dual solution obtained by CEDO is $\mathbf{x}^D = (1.5, 1.5)$ with $\boldsymbol{\lambda}^D = (1, 0)$. The corresponding dual objective value $\theta^D = 0.5$. In this problem, the optimal dual objective value is identical to that of the

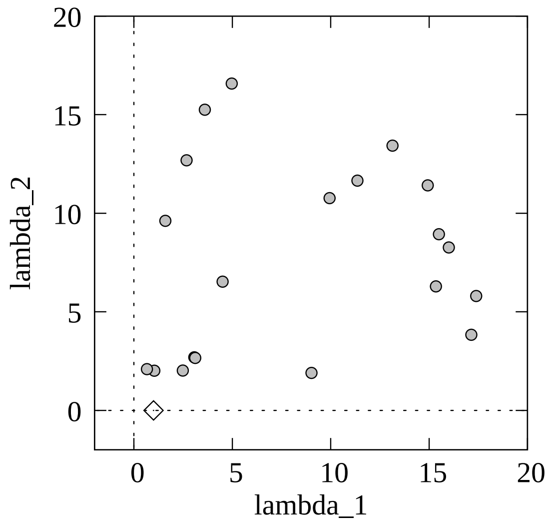


Figure 2: $P_{\boldsymbol{\lambda}}$ population at generation 1 for P1.

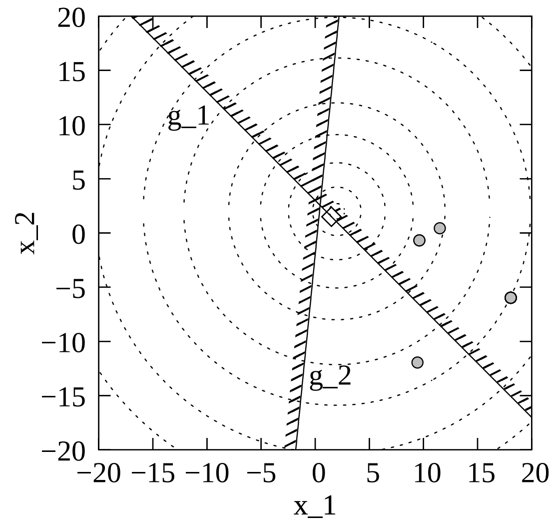


Figure 3: $P_{\mathbf{x}}$ population at generation 1 for P1.

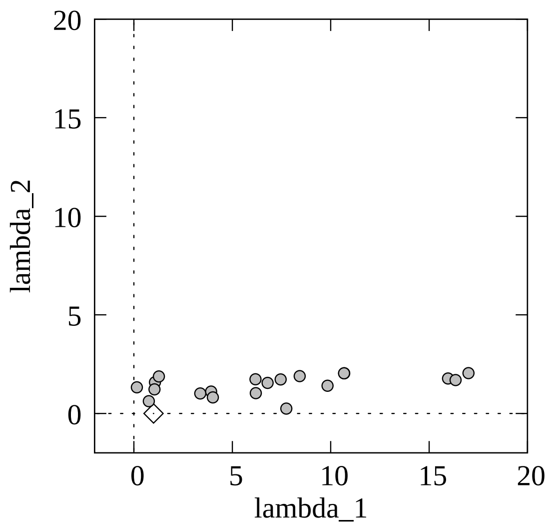


Figure 4: $P_{\boldsymbol{\lambda}}$ population at generation 7 for P1.

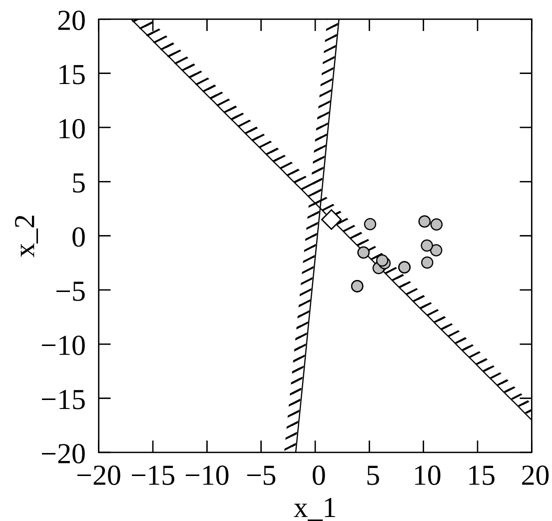


Figure 5: $P_{\mathbf{x}}$ population at generation 7 for P1.

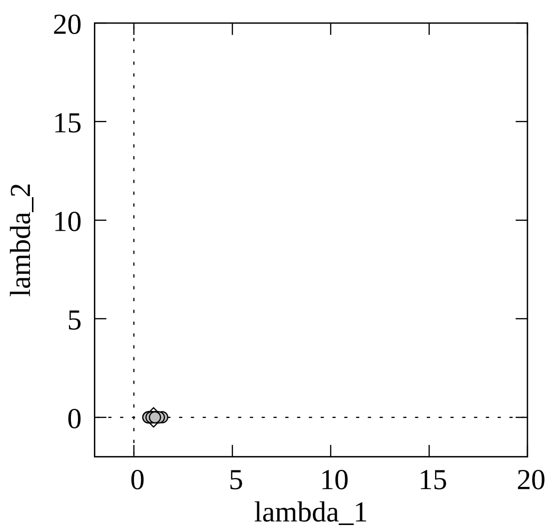


Figure 6: $P_{\boldsymbol{\lambda}}$ population at generation 20 for P1.

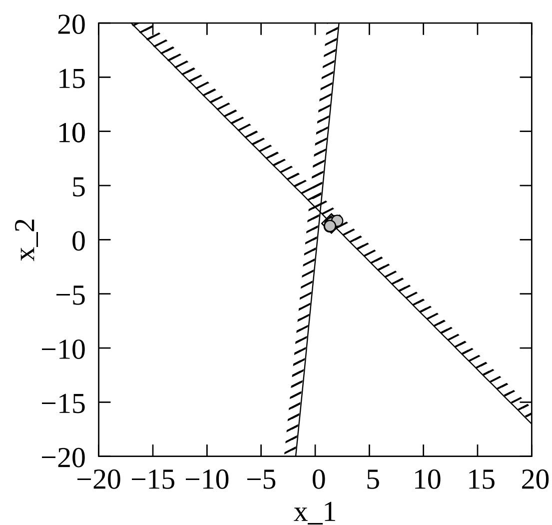


Figure 7: $P_{\mathbf{x}}$ population at generation 20 for P1.

Table 3: Comparison of optimized dual function value and required number of solution evaluations (SE) in 25 runs for zero-duality gap test problems P1 to P6. The column marked with 'S' represents the number of successful runs our of 25 runs. Best results are shown in bold.

			Obtained θ^D			Total SE		
		S	Best	Median	Worst	Best	Median	Worst
P1	CEDO	25	0.5000	0.4998	0.4985	15,892	23,224	34,647
	N-GA	25	0.5000	0.4998	0.4990	5,214	8,386	15,980
	N-Cl	25	0.5000	0.5000	0.5000	86	142	326
P2	CEDO	25	1.0000	1.0000	1.0000	10,658	16,825	26,603
	N-GA	25	1.0000	0.9999	0.9995	8,247	11,453	15,914
	N-Cl	25	1.0000	1.0000	0.9998	353	619	757
P3	**CEDO**	25	0.2500	0.2500	0.2499	32,912	51,195	67,385
	N-GA	25	0.2500	0.2500	0.2499	53,830	62,453	73,739
	N-Cl	0	—	—	—	—	—	—
P4	**CEDO**	22	13.5908	13.5908	13.5905	16,089	201,173	806,751
	N-GA	0	—	—	—	—	—	—
	N-Cl	0	—	—	—	—	—	—
P5	CEDO	25	680.6300	680.6297	680.6291	305,937	1,069,624	2,384,328
	N-GA	25	680.6300	680.6298	680.6293	112,8187	2,932,619	5,589,232
	N-Cl	25	680.6300	680.6300	680.6301	23,310	28,829	43,045
P6	CEDO	25	24.3061	24.3060	24.3058	737,254	933,129	1,238,923
	N-GA	25	24.3061	24.3059	24.3058	1,521,963	2,422,975	3,822,177
	N-Cl	25	24.3062	24.3061	24.3060	38,997	55,137	87,137

primal problem, indicating a zero duality gap. Figure 8 shows the convergence of the values of $L(\mathbf{x}^{\boldsymbol{\lambda}^b}, \boldsymbol{\lambda}^b)$ for the population-best $\boldsymbol{\lambda}^b$ solution and the corresponding $f(\mathbf{x}^{\boldsymbol{\lambda}^b})$ value with the number of generations. Interestingly, the weak duality principle is satisfied by this plot, meaning that for a feasible $\mathbf{x}$ and a feasible $\boldsymbol{\lambda}$, the Lagrangian dual function value is always smaller than that of the primal function value. Another interesting matter to note is that with generations, both these function values come closer and eventually become equal, confirming the property of a zero duality gap problem. The monotonic variations of primal and dual function value with generation counter is also another interesting phenomenon.

Table 3 shows convergence of the coevolutionary algorithm for 25 runs. As mentioned before, the problem P1 is a convex objective function and the feasible decision variable space is also convex. Since the problem is simplistic due to its convexity and low dimensionality properties, the hybrid nested approaches work with a much smaller number of solution evaluations (SE) than the proposed coevolutionary dual optimization approach. The classically nested algorithm (N-Cl) converges very quickly to the true dual solution in this problem. However, as we shall see, for more complex problems, this is hardly the case.

Barbosa's CGA [2] and Jensen's ACGA [24] approaches are tried next starting from an identical variable ranges that were used in CEDO simulations. The initial population used a range of ($x_i \in [-20, 20]$ and $\lambda_i \in [0, 20]$) and thereafter the variables are allowed to vary in the following ranges: $x_i \in [-10000, 10000]$ and $\lambda_i \in [0, 10000]$. Other GA parameter values are also kept identical to that used in CEDO. Despite the problem having a zero duality gap, the CGA approach is not able to find the dual solution in any of the 25 runs satisfying the termination condition described in Section 4.3. The wide range of allowable values of $\mathbf{x}$ and $\boldsymbol{\lambda}$-vectors and inherent noise in lower-level evaluation procedures have caused CGA to not converge adequately to the dual solution. To investigate the working of these methods further, next, we use a tighter range of bounds: $x_i \in [-2, 2]$ in all generations including in the creation of the initial population. The bounds on $\boldsymbol{\lambda}$ is kept the same as before. Now, the CGA approach is able to find the dual solution in 12 of 25 runs (Table 4). Jensen's ACGA approach does not converge to the dual solution according to the termination criterion for the case with a wider variable range mentioned above, however for the tighter variable range, it works 17 of 25 runs. This highlights and agrees with our argument that coevolutionary approaches (symmetric or asymmetric implementations) are noise-prone in solving min-max or dual problems and some local searches in solving the lower-level problem are needed to steer the search towards the right solution. While CEDO takes only 23,224 solution evaluations (even with the

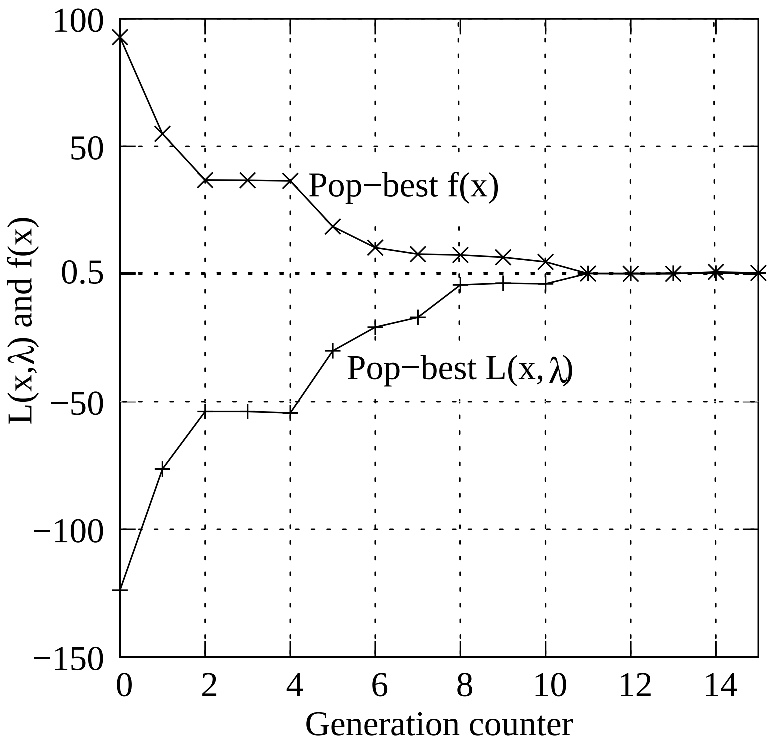


Figure 8: Variation of current best $L(\mathbf{x}^{\boldsymbol{\lambda}^b}, \boldsymbol{\lambda}^b)$ and the corresponding $f(\mathbf{x}^{\boldsymbol{\lambda}^b})$ with the number of generations for Problem P1.

wider variable ranges) on median, as shown in Table 3, CGA and ACGA take 46,620 and 325,505 solution evaluations, respectively (with tighter variable ranges). The use of a local search on a few population members in CEDO helps reduce the associated evaluation noise on all 25 runs.

In every generation of the proposed CEDO algorithm, a few local search operations are performed. Table 5 shows an average of the proportion of population members undergoing local search over generations for problem P1. Less than 15% population members have undergone in any of the 25 runs and in any generation for this problem. Thus, although local search helps to eliminate the evaluation noise, not all population members need to be local-searched for this purpose, as it is done in a nested algorithm. Only a small fraction of population members are needed for this purpose by the proposed CEDO algorithm. It is interesting and noteworthy that still a good compromise between evaluation noise and correct convergence is possible with the proposed approach.

6.1.2 Results on Problem P2

Table 2 shows that the dual solution obtained by the proposed CEDO algorithm is almost identical to the primal solution. Table 3 shows that the CEDO algorithm is comparatively computationally expensive compared to the two nested algorithms. The zero duality gap and low-dimension of the problem makes the problem easier to solve using classical and nested algorithms. Importantly, the proposed CEDO also finds the theoretical dual solution in all 25 runs.

In this problem, CGA and ACGA are able to find all 25 runs converging to the exact dual solution with a tighter range of $x_i \in [-20, 20]$ fixed for all generations (Table 4), but the solution evaluations are (on median) 447,048 and 89,280, respectively, whereas the proposed CEDO takes only 16,825 solution evaluations to achieve the dual solution with a similar termination condition (Table 3) and with a wide range of variable bound ($x_i \in [-10000, 10000]$). The use of local search in CEDO algorithm helps reduce the evaluation noise and as a result hasten the convergence to the dual solution, despite working on a much larger search space.

Table 4: Performance of CGA and ACGA are presented with a tighter variable range for $\mathbf{x}$ for zero duality gap problems. For these tighter variable ranges, both methods performed better, but on the original wider variable range ($x_i \in [-10000, 10000]$) both algorithms were not successful even once.

Algo-rithm	Succ.	$L(\mathbf{x}, \boldsymbol{\lambda})$			Total FE		
		Best	Median	Worst	Best	Median	Worst
Problem P1, Range: $\mathbf{x} \in [-2, 2]$							
CGA	12	0.5000	0.4997	0.4955	46,620	164,045	468,720
ACGA	17	0.4999	0.5021	0.5257	16,800	325,505	739,600
Problem P2, Range: $\mathbf{x} \in [-20, 20]$							
CGA	25	1.0012	0.9942	0.1334	71,820	447,048	1,533,420
ACGA	25	1.0000	1.0020	1.0273	23,600	89,280	234,400
Problem P3, Range: $\mathbf{x} \in [0, 1]$							
CGA	23	0.2508	0.2456	0.3021	48,720	134,820	483,420
ACGA	23	0.2502	0.2572	0.4035	16,800	81,286	315,200
Problem P4, Range: $\mathbf{x} \in [-20, 20]$							
CGA	25	13.5910	13.6350	14.7392	65,520	901,908	2,089,920
ACGA	23	13.5911	13.5873	13.3630	34,000	286,678	682,800
Problem P5, Range: $\mathbf{x} \in [-1, 5]$							
CGA	1	679.7388	679.7388	679.7388	452,640	452,640	452,640
ACGA	1	681.1766	681.1766	681.1766	609,600	609,600	609,600
Problem P6, Range: $\mathbf{x} \in [1, 10]$							
CGA	8	24.3049	24.2316	23.7789	3,894,480	18,798,480	32,341,680
ACGA	5	24.3138	23.9328	22.9574	896,000	4,417,280	8,307,200

6.1.3 Results on Problem P3

Next, we move to a problem, in which both the objective function $f(\mathbf{x})$ and the feasible region are non-convex. In addition to the two given constraints, there are four other linear constraints that arise due to variable bounds. At the primal solution, only one of these six constraints ($x_1 \leq 0.5$) is active.

Table 2 shows that the dual solution obtained by CEDO algorithm is close to the theoretical dual solution and there is a zero duality gap. Figure 9 is plotted for the Lagrange function value at the $P_{\boldsymbol{\lambda}}$-best solution (that is, $L(\mathbf{x}^{\boldsymbol{\lambda}^b}, \boldsymbol{\lambda}^b)$). The figure shows that the population-best dual objective value approaches the population-best primal objective value with generations. In this problem, the variations in population-best primal and dual objective values are not monotonic with generations. The non-convexity in the problem does not allow a steady reduction in the gap between these two objective values with generations, however an overall reduction in the gap to finally to zero is evident from the figure.

Table 3 gives a comparison between the results obtained in 25 runs with both the coevolutionary and nested algorithms. Here, the proposed coevolutionary approach performs the best. Although the hybrid nested approach (N-GA) is able to find the dual solution in all 25 runs, none of the 25 runs of the classically nested approach (N-Cl) is able to converge to the dual solution. To investigate this further, in Table 6, we show three of the 25 cases in which N-Cl gets stuck to a sub-optimal solution. It is clear that these solutions are far from the true dual solution. In the wide allowable range of $\mathbf{x}$ ($x_i \in [-10000, 10000]$), the primal objective value $f(\mathbf{x})$ has a much higher magnitude than the constraint values. This needed the $\boldsymbol{\lambda}$ values to be large to have a significant effect on the Lagrangian function. However, the optimal dual solution requires a value of one for λ_4 and zero for other λ_i values. As a result, the `fmincon()` routine is unable to optimize the lower-level problem adequately to solve the overall dual problem and get stuck to some arbitrary values of $\boldsymbol{\lambda}$, as shown in Table 6. The population approach of evolutionary algorithm is able to overcome the scaling problem in this problem and every time converges to the true dual solution.

Both CGA and ACGA could not locate the dual solution in any of the 25 runs. However, with an extremely tight variable range ($x_i \in [0, 1]$ kept throughout), 23 of 25 runs are able to locate the dual solution. Table 4 shows that even with a tighter range CGA requires 134,820 solution evaluations (median of 25 runs), whereas ACGA takes 81,286 solution evaluations. The asymmetric treatment of upper and lower-level subproblems in ACGA helps to reduce the number of solution evaluations. Interestingly, as presented in Table 3, the proposed CEDO takes only 51,195 solution evaluations, thereby emphasizing the importance of local search in a faster convergence to the dual solution.

Table 5: Local Search statistics in 25 runs for the zero and finite duality gap test problems with CEDO algorithm.

	Proportion of $P_{\boldsymbol{\lambda}}$ for Local Search per generation		
	Best	Median	Worst
Zero Duality Gap Problems			
P1	0.1127	0.1279	0.1437
P2	0.1230	0.1717	0.2006
P3	0.0639	0.0823	0.0984
P4	0.0719	0.0806	0.0951
P5	0.0432	0.0462	0.0503
P6	0.0335	0.0384	0.0453
Finite Duality Gap Problems			
P7	0.0652	0.0784	0.1077
P8	0.0186	0.0197	0.0209
P9	0.0406	0.0518	0.0703

Table 6: Three different solutions in which N-Cl algorithm gets stuck for problem P3.

Run	$\boldsymbol{\lambda}$	$\mathbf{x}$	$L(\mathbf{x}, \boldsymbol{\lambda})$
1	(0.0013,0.0000,0.0000,0.5981,0.0000,0.0000)	(0.7009,0.4913)	0.2248
2	(8.7199,0.0178,10.8741,8.7065,8.4074,6.6067)	(39.0441,1522.8661)	6174.5450
3	(0.0000,0.0000,6.7355,0.0000,0.0000,3.3925)	(0.9944,0.9718)	10.1322

As depicted in Table 5, about 5% population members were needed to be local searched for this problem. The table shows that as the problems get more complex, lesser proportion of population members are needed to be local searched, thereby indicating that although local search helps reduce the evaluation noise, only a small fraction of population members is enough for this purpose to constitute a good search algorithm.

6.1.4 Results on Problem P4

Figure 10 shows a sketch of the feasible region and it is clear that that the feasible region is non-convex for P4. As shown in Figure 10, this problem has four optima having identical objective values (shown by ‘x’), however none of them is feasible. Thus, although there is a single constrained minimum solution, there are four attractors for the objective function. Moreover, the feasible region is non-convex and narrow. These complexities make the task of the nested search difficult to converge to the feasible region. For this problem, even the hybrid nested approach (N-GA) is not able to converge to the dual solution in any of the 25 runs. However, the proposed coevolutionary approach is able to find the true dual solution in 22 out of 25 times. Figure 11 shows the variation of $L(\mathbf{x}^{\boldsymbol{\lambda}^b}, \boldsymbol{\lambda}^b)$ and $f(\mathbf{x}^{\boldsymbol{\lambda}^b})$ with the number of generations. The complexity of the problem is evident from the non-smooth variation of both primal and dual objective values. The difference between the coevolutionary approach and the nested algorithms is clear from this example. In the presence of multi-modality in the objective function, the overall success of the nested procedure largely depends in the accuracy of the lower-level optimization task. The use of the coevolutionary approach becomes beneficial in such problems.

In this problem, both CGA and ACGA is able to find the dual solution with a tighter range ($x_i \in [-20, 20]$) of variable bounds. The performance of ACGA is better than that of CGA, as can be seen from Table 4. The performance of CEDO (median solution evaluations of 201,173) is better than that of ACGA (286,678).

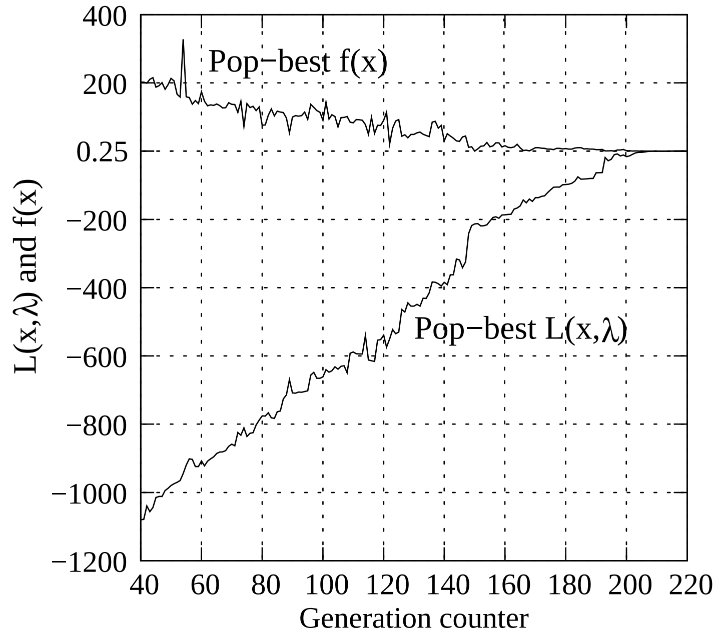


Figure 9: Variation of current best $L(\mathbf{x}^{\boldsymbol{\lambda}^b}, \boldsymbol{\lambda}^b)$ and the corresponding $f(\mathbf{x}^{\boldsymbol{\lambda}^b})$ with the number of generations for Problem P3.

6.1.5 Results on Problem P5

This problem has a zero duality gap, but has enough complexity for the hybrid nested approach (N-GA) to not work well compared to our proposed coevolutionary approach. It is clear that due to the nested nature of evolutionary optimization tasks having a population of solutions to compute in each generation, the N-GA takes relatively more solution evaluations to get an identical accuracy in the dual solution. However, the nested classical algorithm (N-Cl) is able to quickly find the dual solution in all 25 runs. In this problem, a point-by-point optimization approach works and when this does, it is difficult for a population-based optimization algorithm to compete.

This problem is found to be difficult for CGA and ACGA. Only with a tight range of variable bounds ($\mathbf{x} \in [-1, 5]$), one out of 25 runs is able to converge to the dual solution in 452,640 and 609,600 solution evaluations, respectively. Although CEDO took 1,069,624 solution evaluations, it could find the dual solutions by using a much wider range of variables ($\mathbf{x} \in [-10000, 10000]$) and importantly in 25 out of 25 runs, thereby making CEDO algorithm a more reliable approach.

6.1.6 Results on Problem P6

The conclusions derived from the results on this problem is similar to that in problem P5. All three algorithms – CEDO, N-GA and N-Cl – are successful in finding the dual problem here, but the classically nested algorithm (N-Cl) is better than the two evolutionary approaches. Both the coevolutionary and nested algorithms are able to find the dual solution in all 25 runs.

CGA and ACGA are able to find the dual solution only with a reduced variable range ($x_i \in [1, 10]$) and requiring 18,798,480 and 4,417,280 solution evaluations, respectively. CGA converged in 8 out of 25 times and ACGA converged in 5 out of 25 runs. On the contrary, CEDO converged to the dual solution in all 25 out of 25 runs and with a median solution evaluations of 933,129. These results are remarkable and show the usefulness of local search in reducing evaluation noise.

6.2 Finite Duality Gap Problems

As we demonstrate in this section, the next three problems (P7-P9) possess a finite duality gap. In these problems, the optimal dual function value is smaller than the optimal primal objective value. We present

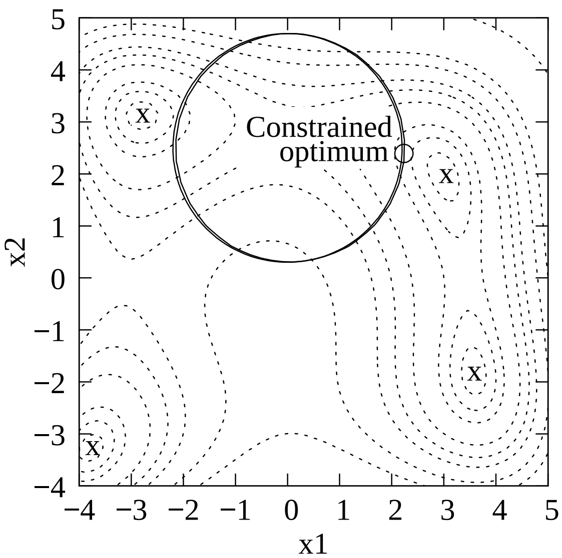


Figure 10: Feasible search space lies within the two circles in Problem P4. Four minima of the objective function are infeasible attractors of this problem.

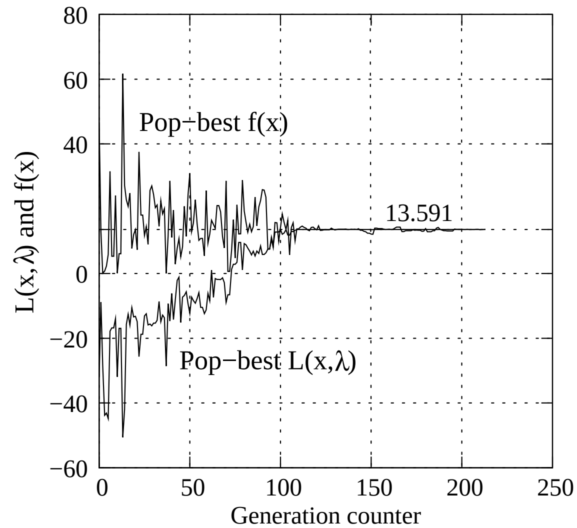


Figure 11: Variation of current best $L(\mathbf{x}^{\boldsymbol{\lambda}^b}, \boldsymbol{\lambda}^b)$ and the corresponding $f(\mathbf{x}^{\boldsymbol{\lambda}^b})$ with the number of generations for Problem P4.

results of our coevolutionary approach on these problems in Table 7. It is clear that in all three problems, CEDO is able to find a solution to close to the true dual solution ($\boldsymbol{\lambda}^D$ and corresponding $\theta^D = L(\mathbf{x}^D, \boldsymbol{\lambda}^D)$).

Table 8 presents a comparison of the performance of the three algorithms on these problems. Both nested algorithms fail to find the true dual solution in all these problems. In the following subsections, we discuss further on the performance of CEDO on these problems one at a time.

6.2.1 Results on Problem P7

The optimal primal solution to this problem is $x^P = \pm 0.5$ with $f^P = -0.234375$. The constraint function is non-differentiable for this problem, hence we attempt to find the dual solution graphically, instead of mathematically. In the dual formulation given in Equation 3, the dual solution λ^D would maximize $\theta(\lambda)$, where $\theta(\lambda)$ corresponds to the minimum $L(x, \lambda)$ with respect to x. For P7, Figure 12 shows the variation of $L(x, \lambda)$ for four different values of λ. Let us first consider $\lambda = 0$. The minimum of $L(x, 0)$, as can be seen from the figure, is $x^D = \sqrt{2} = \pm 1.414$. For this point, $L(x^D, 0) = -1$. Now, we observe that as we increase λ value from zero, the minimal solution x^D gets closer to the feasible region and $L(x^D, \lambda)$ has a higher value. For example, for $\lambda = 0.5$, $x^D = \pm 1.2670$ and $L(x^D, 0.5) = -0.5776$. Since for the dual problem, higher $L(x^D, \lambda)$ is better, we should continue to increase λ for finding the dual solution. At $\lambda = 0.7698$, there are three minima: $x^D = 0, \pm 1.1547$ having $L(x^D, 0.7698) = -0.3849$. When λ is increased further, there is only one minimum point: $x^D = 0$ having $L(x^D, \lambda) = -0.5\lambda$. Thus, for $\lambda > 0.7698$, the optimal $L(x^D, \lambda)$ value decreases, thereby making $\lambda = 0.7698$ a maximum of the Lagrange function. Hence, this λ value is the dual solution for the problem P7, or $\lambda^D = 0.7698$ and $\theta^D = -0.3849$. Figure 13 shows the $\theta(\lambda) = L(x^D, \lambda)$ variation with λ. The duality gap ($\rho = f^P - \theta^D$) present in this problem can be clearly seen from this figure. Also the kink (or non-differentiability) at the maximum point is evident in this problem.

In order to capture all the three dual solutions, 20 population members are chosen for both $P_{\mathbf{x}}$ and $P_{\boldsymbol{\lambda}}$. Table 7 shows the dual solutions obtained with the proposed CEDO algorithm. It can be seen that the algorithm is able to find all the three dual solutions. Both nested algorithms with a classical optimization algorithm employed for solving the lower-level problem are unable to find the dual solution for this problem, possibly due to the non-existence of the derivative of the $\theta(\lambda)$ function. However, Table 8 shows that the CEDO algorithm is able to find the optimal dual solutions in all 25 runs. Figure 14 shows the variation of population-best $L(x^{\lambda^b}, \lambda^b)$ with the generation counter. It is interesting how the proposed CEDO algorithm is able to improve the Lagrange function value with generation and eventually gets settled to the optimal

Table 7: Dual solutions obtained by CEDO for the finite duality gap problems (P7-P9). In some cases, multiple solutions are found by CEDO algorithm in different runs.

		$\boldsymbol{\lambda}^D$	$\mathbf{x}^D$	$L(\mathbf{x}^D, \boldsymbol{\lambda}^D)$
P7	Theory	0.7698	0, $\pm$ 1.1547	-0.3849
	CEDO	0.7675	-1.1555	-0.3864
		0.7746	0.0000	-0.3873
		0.7782	1.1504	-0.3794
P8	Theory	(0, 0, 1, 0, 0, 0, 0, 0)		10
	CEDO	(0.0000, 0.0000, 1.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000)	(10000.0000, -9130.1734, -7849.7641)	10.0000
P9	Theory	(0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)		2100
	CEDO	(0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 1.0000, 0.0000, 1.0000, 0.0000, 1.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000)	(5542.5512, 6150.2213, 9980.6527, -8729.6993, 7038.3150, -7914.8453, 9655.6132, -1585.4457)	2100.0000

dual function value. The optimal primal function value is also marked in the figure to demonstrate the finite duality gap property of this problem.

Since this is an one-variable problem, we take this example to illustrate the working of the proposed coevolutionary algorithm on a finite duality gap problem with the help of snapshots taken at four different generations. Given the dual formulation in Equation 3, the objective $\mathcal{F}_{\boldsymbol{\lambda}}$ is taken as minimization of $-L(x, \lambda)$ and $\mathcal{F}_{\mathbf{x}}$ is taken as minimization of $L(x, \lambda)$. Figure 15 shows the $L(x, \lambda)$ values for each pair of $(P_{\mathbf{x}}, P_{\boldsymbol{\lambda}})$ taken at four different generations. All the values marked by diamonds represent $L(x_i, \lambda_j)$ calculated by taking pairs $({P_{\mathbf{x}}}^i, {P_{\boldsymbol{\lambda}}}^j)$ for all $i \in P_{\mathbf{x}}, j \in P_{\boldsymbol{\lambda}}$. The line indicates $\theta(\lambda_k) = L(x_k^*, \lambda_k)$ calculated by finding the optimal x_k^* for each λ_k. In the coevolutionary algorithm, a partner ${P_{\mathbf{x}}}^j$ is searched for each ${P_{\boldsymbol{\lambda}}}^j$ such that it corresponds to the minimum $\mathcal{F}_{\mathbf{x}}$ in the given population, $P_{\mathbf{x}}$. This partner would hence be the one closest to $L(x_j^*, \lambda_j)$ line and is used to approximate the fitness of the ${P_{\boldsymbol{\lambda}}}^j$ as $L(x_j, \lambda_j)$. As the generation progresses, $P_{\boldsymbol{\lambda}}$ evolves closer to the optimal λ value taking help from $P_{\mathbf{x}}$ to provide a near optimal solution at x and from the local search operator to improve the approximated fitness of the solutions better than the current best at every generation. Figure 15(b) shows that initial population does not have any λ close to the optimal dual value and how from an initial large diverse set of points the population $P_{\boldsymbol{\lambda}}$ progresses towards the optimal region with generations (Figures 15(a) to (d)).

CGA and ACGA approaches are also applied to this finite duality gap problem. Both these methods could not find the dual solution in any of the 25 runs, despite the problem being a single-variable problem. Table 8 shows that when the bound on x is reduced to $[-2, 2]$, ACGA works 23 out of 25 times with a median solution evaluations of 326,313. However, even with the wider range of $x \in [-10000, 10000]$, CEDO is able to converge to one of the dual solutions in 20 out of 25 times with a median solution evaluations of only 49,325. CGA could not converge even with the reduced variable range.

6.2.2 Results on Problem P8

The derivation of the dual solution is not straightforward for this problem, but can be made with some analysis of the corresponding Lagrangian function. We presented our arguments to come up with the dual solution (presented in Table 7) in the Appendix, but here we simply state the solution: $\boldsymbol{\lambda}^D = (0, 0, 1, 0, 0, 0, 0, 0)$ and $\theta^D = 10$. Since the optimal primal function value is $f^P = 110$, there is a finite duality gap of $\rho = f^P - \theta^D = 100$.

Table 8: Comparison of optimized dual function value and required number of solution evaluations (SE) in 25 runs for finite duality gap test problems P7 to P9. S denotes number of successful runs out of 25 runs. For problem P7, CGA and ACGA results (indicated with a *) are shown for $x_i \in [-2, 2]$; for other two cases, $x_i \in [-10000, 10000]$.

Algo-rithm	S	θ^D Best	θ^D Median	θ^D Worst	Total SE Best	Total SE Median	Total SE Worst
Problem P7							
CEDO	20	−0.3864	−0.3861	−0.3558	8,812	49,325	158,329
CGA*	0	—	—	—	—	—	—
ACGA*	23	-0.3849	-0.3740	-0.2514	8,400	326,313	775,200
Problem P8							
CEDO	24	10.0000	10.0002	10.0149	160,255	399,131	979,096
CGA	24	10.0000	10.0016	10.0275	219,480	558,348	1,572,630
ACGA	23	10.0000	10.0033	10.0779	153,000	376,708	1,209,600
Problem P9							
CEDO	5	2100.0000	2100.2740	2101.2555	2,116,991	5,262,194	7,990,475
CGA	1	2100.7121	2100.7121	2100.7121	3,959,280	3,959,280	3,959,280
ACGA	21	2100.0000	2100.0001	2100.0031	2,240,000	3,710,171	7,494,400

Table 8 shows that the proposed CEDO algorithm finds the true dual solution in 24 out of the 25 runs performed, whereas both the nested algorithms fail to find in any of the 25 runs. Variation in θ^D values over 25 runs from different initial populations is not that significant. Figure 16 shows the variation of the population-best $L(\mathbf{x}^b, \boldsymbol{\lambda}^b)$ with the generation counter. The optimal primal objective value is also marked in the figure. It is clear that the obtained dual function value is smaller than the optimal primal objective value, thereby demonstrating a finite duality gap.

Interestingly, both CGA and ACGA work well for this problem with the original variable ranges (Table 8). In fact, ACGA takes slightly smaller number of solution evaluations (376,708 as opposed to 399,131) than the CEDO approach. Somehow, in this problem, the computations in local search is larger than additional computations needed to counter the evaluation noise for lower-level population members.

As shown in Table 5, about 2% population members are local searched in any generation of all 25 runs. This indicates that the CEDO algorithm utilizes lower-level optimal solution information only for a few of population members to converge to the dual solution.

6.2.3 Results on Problem P9

For this problem as well, we obtain the dual solution by analyzing the Lagrangian function. The arguments towards achieving the dual solution are presented in the Appendix. Here, we simply present the dual solution: $\lambda_j^D = 0, \ \forall \ j \in \{1, \ldots, 22\} \setminus \{7, 9, 11\}$ and is equal to 1, $\forall \ j \in \{7, 9, 11\}$. The optimal dual function value is $\theta^D = 2100$. Since the optimal primal function value is $f^P = 7049.3309$, there is a finite duality gap of $\rho = 4949.3309$ in this problem.

Table 7 shows the dual solution obtained using the CEDO algorithm. The dual solution is obtained with a finite duality gap of 4949.3303. Both nested algorithms fail to find the dual solution in all 25 runs. Variation in the obtained θ^D value over 25 runs are not that significant. This problem is reported to be a difficult problem in the EA literature [12]. However, Table 8 shows that the proposed coevolutionary algorithm is able to find the dual solution in 5 out of 25 runs for this problem. Figure 17 shows the variation of the population-best $L(\mathbf{x}^{\boldsymbol{\lambda}^b}, \boldsymbol{\lambda}^b)$ value. The optimal primal objective value is also marked in the figure. The duality gap can be clearly seen from this figure.

Like in problem P8, in this problem as well, ACGA performs better than CEDO. For this problem, a small proportion (less than about 7%) population members were needed to be local searched as shown in Table 5, thereby indicating that the proposed approach uses local search only sparingly in converging to the dual solution.

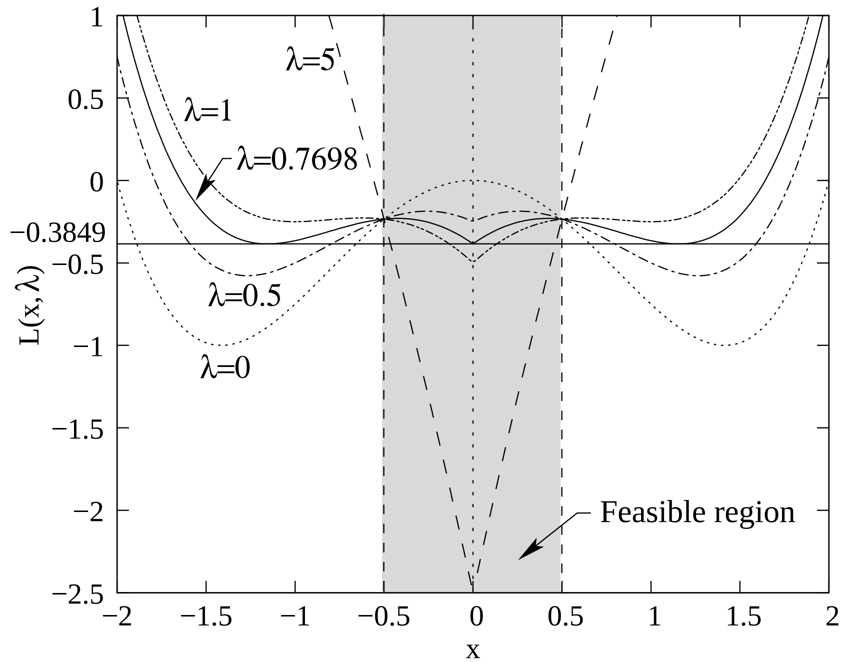


Figure 12: Plot of Lagrangian function for different values of λ for problem P7.

6.3 Infinite Duality Gap Problems

The remaining two test problems (P10 and P11) are found to exhibit an infinite duality gap. In these problems, the optimal dual function value is negative infinity. Table 9 shows the dual solutions obtained by the proposed coevolutionary dual optimization algorithm on these problems.

Table 9: Solutions obtained for best Lagrangian value in infinite duality gap problems given in Section 6.3.

		$\boldsymbol{\lambda}^D$	$\mathbf{x}^D$	θ^D
P10	Theory	$(0,\infty,\infty,0,\infty,0)$	$(-\infty,-\infty)$	$-\infty$
	CEDO	(0, 10000, 10000, 0, 10000, 0)	$(-10000, -10000)$	$-6.6151(10)^9$
P11	Theory		$x_1x_5 = -\infty$	$-\infty$
	CEDO	(0.0025, 0.0015, 0.0166, 0.0031, 1.0780, 10.5412, 0.0006, 0.1033, 3.8817, 3.4728, 10000, 7.1913, 0, 0.0001, 0.4155, 8871.0116)	(10000, −10000, 990.1478, −10000, −10000)	$-1.7721(10)^8$

For these problems, since the dual function value tends to become a large negative value with generations, no comparison is made with nested or CGA and ACGA approaches.

6.3.1 Results on Problem P10

We attempt to compute the dual solution by writing the Lagrange function:

$$\begin{aligned} L(\mathbf{x},\boldsymbol{\lambda}) \;=\; & (x_1-10)^3+(x_2-20)^3+\lambda_1(-(x_1-5)^2-(x_2-5)^2+100)+ \\ & \lambda_2((x_1-6)^2+(x_2-5)^2-82.81)+\lambda_3(13-x_1)+\lambda_4(x_1-100)+ \\ & \lambda_5(-x_2)+\lambda_6(x_2-100). \end{aligned}$$

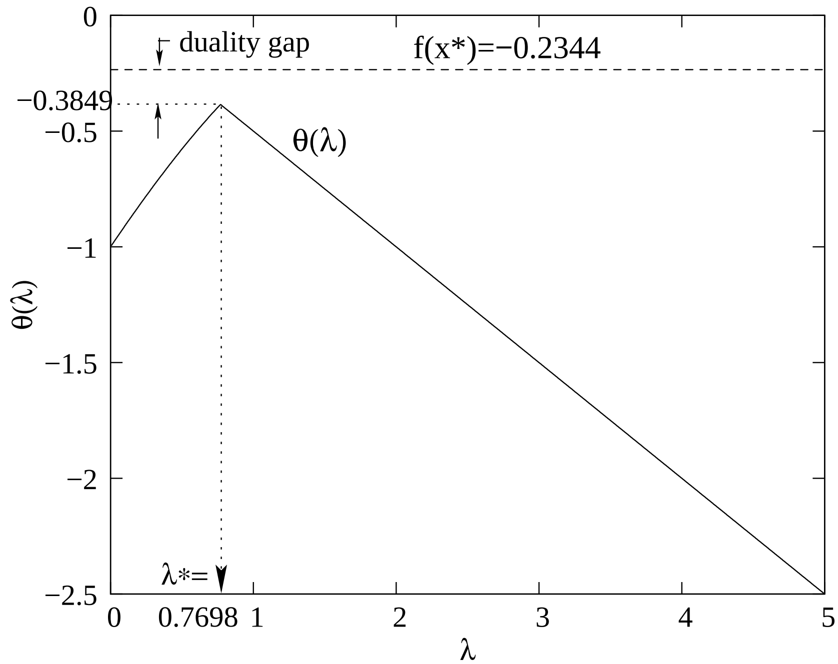


Figure 13: Variation of $\theta(\lambda)$ with λ is shown for problem P7.

It can be seen that the above function is separable in terms of x_1 and x_2. Thus, for a given $\boldsymbol{\lambda}$-vector, $\theta(\boldsymbol{\lambda})$ can be found by minimizing the expressions for x_1 and x_2 separately. Since the variable bounds are treated as constraints here, the infimum of $L(\mathbf{x}, \boldsymbol{\lambda})$ must have to be performed in $\mathbf{x} \in \Re^2$. For the infimum of the above term, cubic terms govern the magnitude. It is clear that for any given $\boldsymbol{\lambda}$-vector, the infimum of $L(\mathbf{x}, \boldsymbol{\lambda})$ will occur for $x_i = -\infty$ for $i = 1, 2$. Thus, $\mathbf{x}^D = (-\infty, -\infty)$. However, due to the involvement of quadratic terms with λ_i values, the maximum of $\theta(\boldsymbol{\lambda})$ will take place for $\lambda_i = 0$ if the multiplier to λ_i is negative and $\lambda_i = \infty$ if the multiplier is positive. Thus, $\boldsymbol{\lambda}^D = (0, \infty, \infty, 0, \infty, 0)$. The corresponding θ^D will be $-\infty$. Since $f^P = -6961.7511$, the problem has an infinite duality gap.

Table 9 shows that both x_1 and x_2 go to their respective lower bound values (-10000) used in the simulation runs. A large negative value of obtained θ^D also indicates the infinite duality gap associated with this problem. For this problem, all 25 runs found the same dual solution. Since the lower and upper bounds of (-10000 and 10000) used here for x_i and λ_i are artificial and the proposed CEDO algorithm is likely to attain the respective bounds if they are changed, we do not report the number of solution evaluations here. In any case, both nested approach were not able to find the dual solution.

6.3.2 Results on Problem P11

The Lagrange function for this problem can be written as follows:

$$
\begin{aligned}
&L(\mathbf{x}, \boldsymbol{\lambda}) \\
&= \; 5.3578547x_3^2 + 0.8356891x_1x_5 + 37.293239x_1 - 40792.141 \\
&\quad - \lambda_1(85.334407 + 0.0056568x_2x_5 + 0.0006262x_1x_4 - 0.0022053x_3x_5) + \\
&\quad \lambda_2(85.334407 + 0.0056568x_2x_5 + 0.0006262x_1x_4 - 0.0022053x_3x_5 - 92) - \\
&\quad \lambda_3(80.51249 + 0.0071317x_2x_5 + 0.0029955x_1x_2 + 0.0021813x_3^2 - 90) + \\
&\quad \lambda_4(80.51249 + 0.0071317x_2x_5 + 0.0029955x_1x_2 + 0.0021813x_3^2 - 110) - \\
&\quad \lambda_5(9.300961 + 0.0047026x_3x_5 + 0.0012547x_1x_3 + 0.0019085x_3x_4 - 20) + \\
&\quad \lambda_6(9.300961 + 0.0047026x_3x_5 + 0.0012547x_1x_3 + 0.0019085x_3x_4 - 25) + \\
&\quad \lambda_7(78 - x_1) + \lambda_8(x_1 - 102) + \lambda_9(33 - x_2) + \lambda_{10}(x_2 - 45) + \lambda_{11}(27 - x_3) + \\
&\quad \lambda_{12}(x_3 - 45) + \lambda_{13}(27 - x_4) + \lambda_{14}(x_4 - 45) + \lambda_{15}(27 - x_5) + \lambda_{16}(x_5 - 45).
\end{aligned}
$$

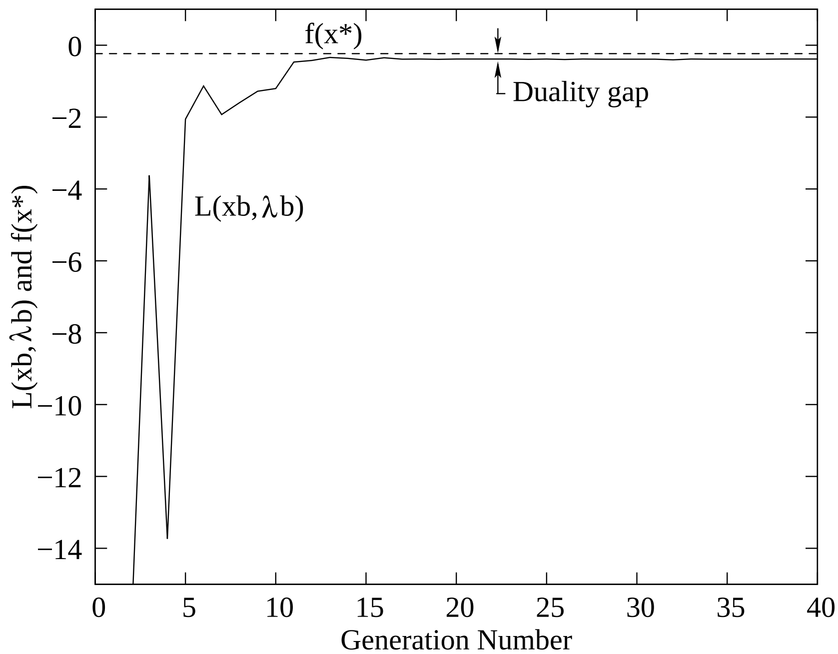


Figure 14: Variation of population-best $L(x^{\lambda^b}, \lambda^b)$ with the generation number for problem P7.

Rearranging the terms, we obtain the following:

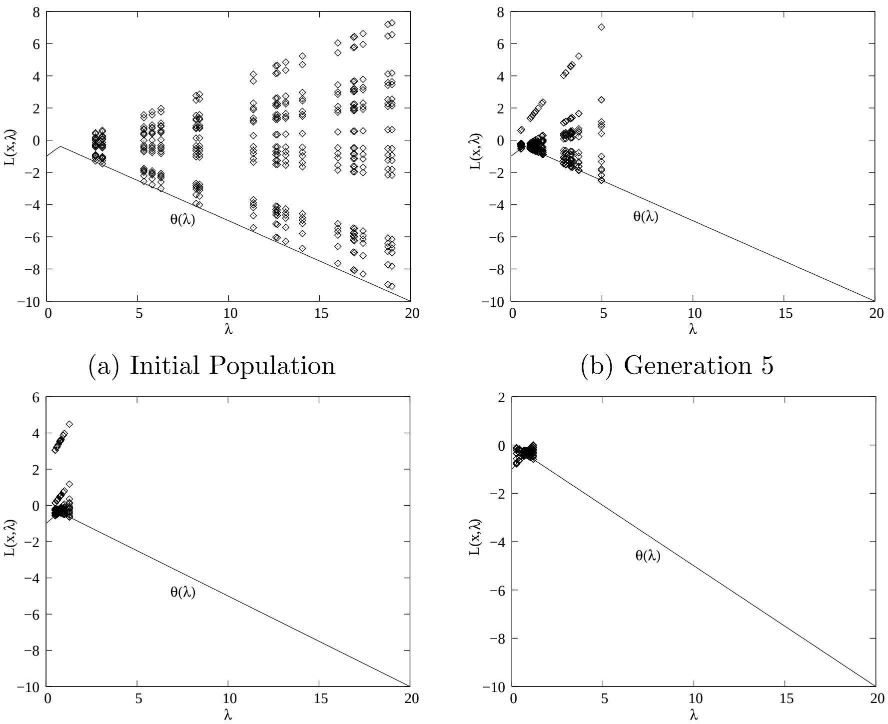
$$
\begin{aligned}
&L(\mathbf{x}, \boldsymbol{\lambda}) \\
&= (37.293239 - \lambda_7 + \lambda_8)x_1 + (\lambda_{10} - \lambda_9)x_2 + (\lambda_{12} - \lambda_{11})x_3 + (\lambda_{13} - \lambda_{14})x_4 + \\
&\quad (\lambda_{16} - \lambda_{15})x_5 + (5.3578547 - 0.0021813\lambda_3 + 0.0021813\lambda_4)x_3^2 + \\
&\quad 0.8356891x_1x_5 + (-0.0056568\lambda_1 + 0.0056568\lambda_2 - \\
&\quad 0.0047026\lambda_5 + 0.0047026\lambda_6)x_3x_5 + 0.0029955(\lambda_4 - \lambda_3)x_1x_3 + \\
&\quad 0.0019085(\lambda_6 - \lambda_5)x_3x_4 - 40792.141 - 92\lambda_2 + 9.48751\lambda_3 - 29.48751\lambda_4 + \\
&\quad 10.699039\lambda_5 - 15.699039\lambda_6.
\end{aligned}
$$

It is interesting to note that for a given $\boldsymbol{\lambda}$-vector, the above function is a quadratic polynomial in $\mathbf{x}$-vector. It can be observed that the coefficient of x_1x_5 is independent of $\boldsymbol{\lambda}$. Since x_i can take any value in the real space, for any given $\boldsymbol{\lambda}$, the infimum of $L(\mathbf{x}, \boldsymbol{\lambda})$ will be $-\infty$ resulted from the positive and negative ∞ value of x_1 and x_5. Since this is true for any $\boldsymbol{\lambda}$-vector, the dual function value will also be $-\infty$. Since $f^P = -30665.5$, this will cause the problem to have an infinite duality gap.

Table 9 shows the dual solution obtained by CEDO algorithm chooses x_1 and x_5 to their allowable upper and lower bounds, respectively, so that their product becomes a large negative number. The obtained $\boldsymbol{\lambda}^D$-vector is also shown. Importantly, the θ^D value is a large negative number, thereby supporting the infinite duality gap property of this problem. Both nested approach were not able to find the dual solution for this problem as well.

6.4 Computational Advantage of CEDO Over the Hybrid Nested Approach

Here we investigate the scenarios when the proposed coevolutionary dual optimization (CEDO) algorithm will be better than the hybrid nested approach (N-GA). Let us assume that in both cases, we have identical population size N. Say, N-GA takes t_n generations and CEDO takes t_c generations of the upper-level task. Also, let us assume that the local search operation in both approaches require on an average S solution evaluations to solve the Lagrange subproblem to the desired accuracy. Then, N-GA requires a total of NSt_n solution evaluations. However, each upper-level generation of CEDO first requires to compute all N^2 solution evaluations (every combination of upper and lower-level variables) to identify population-best $\boldsymbol{\lambda}$-vector. Thereafter, depending on the local search outcome of population-best $\boldsymbol{\lambda}$-vector, a few more local searches may have to be done. Say, on an average, a fraction of l of the population (total lN population



(a) Initial Population (b) Generation 5

(c) Generation 15 (d) Generation 50

Figure 15: Plot of $L(x^i, \lambda^j)$ $\forall$ (i, j) of $P_{\boldsymbol{\lambda}}$ and $P_{\mathbf{x}}$ taken at four different generations for Problem P7.

members) is needed to be local searched in every generation. Thus, the total number of solution evaluations for CEDO algorithm is $(N^2 + lNS)t_c$. As can be seen from Table 5, on nine problems with zero and finite duality gaps, the maximum proportion of local searches at any generation is not more than around 20%. Interestingly, on more difficult problems the proportion of population members being local searched is smaller and is lesser than 10%.

Thus, the CEDO algorithm will be computationally better than N-GA when the following condition is true:

$$\begin{aligned} NSt_n &> (N^2 + lNS)t_c, \\ \text{or, } S &> \frac{N}{(t_n - lt_c)/t_c}. \end{aligned}$$

The quantity lt_c indicates the total number of local searches performed in CEDO approach and t_n indicates the same for N-GA approach. Thus, if CEDO takes more local searches to find the dual solution (that is, $lt_c > t_n$), CEDO is never computationally better than N-GA. But, if CEDO demands less number of local searches than N-GA, CEDO will be computationally more efficient, if the average number of solution evaluations involved in a local search operation is more than the right side term. For equivalent values of t_n and t_c, the right side reduces to $N/(1 - l)$. In most of the complex problems, a maximum of 10% population members were local searched. Thus, in such a scenario, if $S > 1.11N$, in other words, the average number of solution evaluations in a local search is more than 1.11 times the chosen upper-level population size, CEDO will be computationally better. Since a local search operator usually uses gradients of objectives and constraints which when computed numerically requires $2n$ solution evaluations by the central difference method [9], the required number of solution evaluations in a local search operation for a reasonably large and complex problem can be more than the population size chosen for the evolutionary algorithm.

Even for smaller sized problems in this study, we have witnessed a large computational advantage by CEDO over N-GA. This is due to the fact that the number of generations t_n needed by N-GA is also more than that needed by CEDO. What we have also observed in our simulation studies is that in many problems

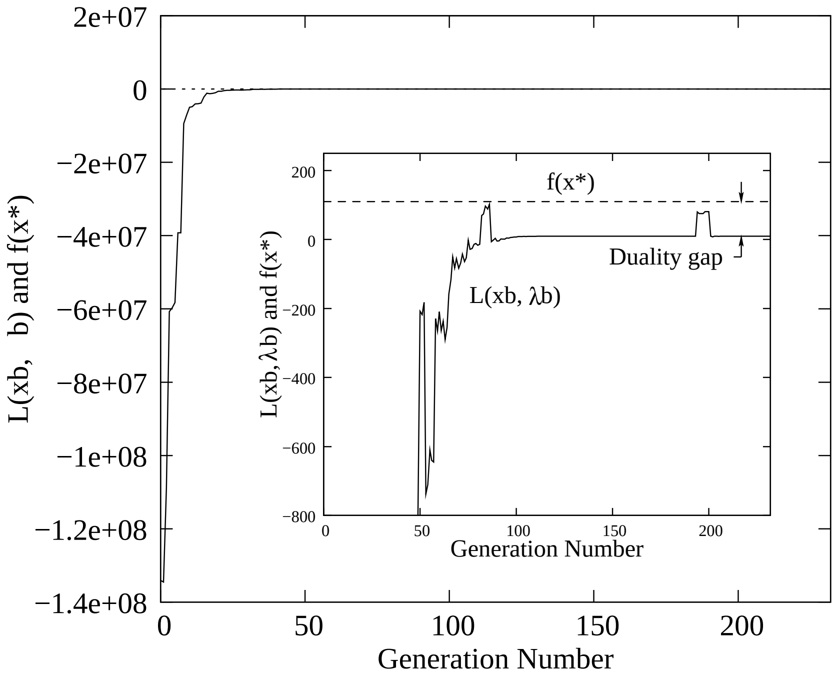


Figure 16: Variation of population-best $L()$ with generation number is shown for problem P8. The optimal primal objective value is also marked. The inset figure shows the details.

computation is not an issue with the nested approaches, rather both N-GA and N-Cl were not able to find the dual solution despite allowing them to be run for a large number of generations and iterations.

7 Conclusions

In this paper, we have suggested a coevolutionary algorithm for solving Lagrangian dual problems derived from a given primal problem. The dual problem is a bilevel optimization problem in which lower-level problem involves finding infimum of the Lagrange function with respect to $\mathbf{x}$-vector and for a fixed $\boldsymbol{\lambda}$-vector, and the upper-level problem involves maximizing the Lagrange function with respect to $\boldsymbol{\lambda}$-vector. The proposed coevolutionary algorithm uses two populations, one for primal variables $\mathbf{x}$ and another for dual variables $\boldsymbol{\lambda}$. Unlike other coevolutionary approaches used for a similar task, our approach is different in that the two evolving populations are not treated equally, rather the evolution of the lower-level population members are more influenced by the evolution of upper-level population members, and not the other way around. Since the solution of a bilevel problem is not symmetric and must have this property, our proposed CEDO algorithm is able to find the dual solution to 11 constrained test problems borrowed from the evolutionary algorithm literature. The difficulty in solving 7 out of 11 chosen test problems can be attributed to the fact that two nested optimization algorithms involving classical and genetic algorithms have not been to solve them.

The primal to dual formulation and their relationship allows one to classify constrained problems into three categories: (i) problems having zero duality gap, (ii) problems having finite duality gap and (iii) problems having infinite duality gap. Not only that this paper is able to find the dual solution in all 11 test problems depicting different types of duality gaps, the paper, for the first time, classified some commonly-used constrained test problems according to their appropriate duality classes. Although a direct correlation between duality gap and difficulty of solving the primal or the dual problem is not ascertained or demonstrated in the optimization literature, here, we have observed that nested algorithms have shown their vulnerability in solving corresponding dual problem for non-zero duality gap problems. However, our proposed coevolutionary approach has worked well on all 11 test problems irrespective of extent of duality gap. In comparison with a couple of existing coevolutionary algorithms, it has been observed that they

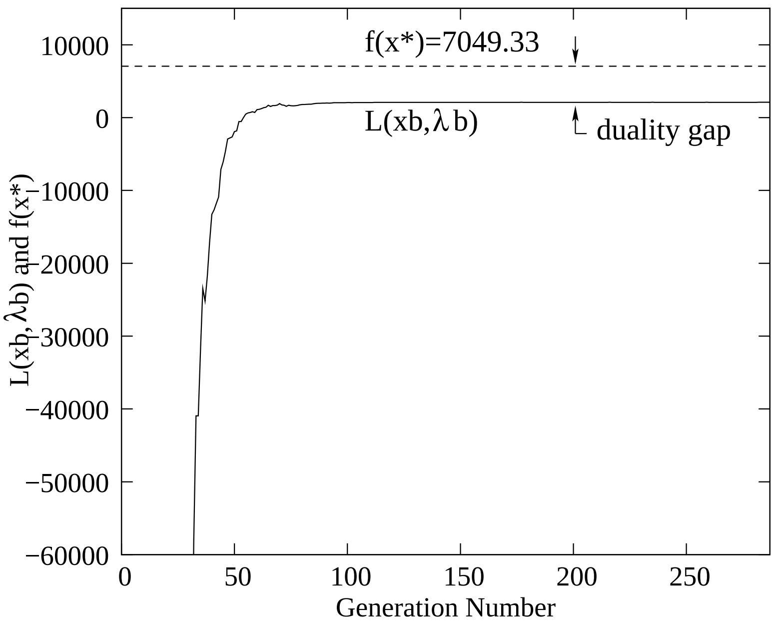


Figure 17: Variation of current population-best $L()$ with the generation number is shown for problem P9. The optimal primal objective value is also marked.

can find the dual solutions when the allowable search space (particularly in $\mathbf{x}$-space) for them is reduced substantially. Of them, Jensen's [24] approach is better and has worked well in solving finite duality gap problems, in general.

The proposed CEDO algorithm is, in principle, capable of solving primal problems, as well, as the primal problem can be converted into an equivalent min-max problem. On a similar account, the proposed CEDO algorithm should also be used to solve generic min-max or max-min problems.

As mentioned above, the dual problems are bilevel in nature. However, dual problems are special bilevel problems in which objective functions in both upper and lower levels are identical. A generic bilevel problem may have different functions in its upper and lower levels. The success of the proposed CEDO algorithm may now be extended to solve generic bilevel problems using coevolutionary means. To make the solution of dual problems practical and computationally fast, surrogate-assisted approaches similar to an earlier study [37] can be tried with our proposed coevolutionary approach. The possibility of coevolving two or more populations representing different types of variables in solving a problem certainly broadens the scope of evolutionary computation field to optimization research and provides it with an unique edge over its traditional counterparts. Starting with Barbosa's 1996 study and following up with other coevolutionary studies, our proposal of an efficient coevolutionary algorithm and successful results on standard constrained test problems for solving dual problems reported here remains as the right step towards taking advantage of coevolution in optimization research.

acknowledgements

Authors wish to thank the Academy of Finland for the support of this work under project number 133387.

References

[1] Abass, S.A.: Bilevel programming approach applied to the flow shop scheduling problem under fuzziness. Computational Management Science **4**(4), 279–293 (2005)

[2] Barbosa, H.J.C.: A genetic algorithm for min-max problems. In: Proceedings of the First International Conference on Evolutionary Computation and Its Application (EvCA'96), pp. 99–109 (1996)

[3] Barbosa, H.J.C.: A coevolutionary genetic algorithm for constraint optimization. In: Proceedings of the 1999 Congress on Evolutionary Computation (CEC-99), pp. 1605–1611 (1999)

[4] Bard, J.F.: Practical Bilevel Optimization: Algorithms and Applications. The Netherlands: Kluwer (1998)

[5] Bazaraa, M.S., Sherali, H.D., Shetty, C.M.: Nonlinear programming: Theory and algorithms. Singapore: Wiley (2004)

[6] Bianco, L., Caramia, M., Giordani, S.: A bilevel flow model for hazmat transportation network design. Transportation Research. Part C: Emerging technologies **17**(2), 175–196 (2009)

[7] Bradley, S., Hax, A., Magnanti, T.: Applied Mathematical Programming. Addison-Wesley (1977)

[8] Branke, J., Rosenbusch, J.: New approaches to coevolutionary worst-case optimization. In: Proceedings of the Parallel Problem Solving from Nature Conference (PPSN-X), (LNCS 5199), pp. 144–153 (2008)

[9] Deb, K.: Optimization for Engineering Design: Algorithms and Examples. New Delhi: Prentice-Hall (1995)

[10] Deb, K.: Multi-objective optimization using evolutionary algorithms. Wiley, Chichester, UK (2001)

[11] Deb, K., Agrawal, R.B.: Simulated binary crossover for continuous search space. Complex Systems **9**(2), 115–148 (1995)

[12] Deb, K., Datta, R.: A fast and accurate solution of constrained optimization problems using a hybrid bi-objective and penalty function approach. In: Proceedings of the IEEE World Congress on Computational Intelligence (WCCI-2010), pp. 165–172 (2010)

[13] Deb, K., Sinha, A.: Solving multi-objective bilevel optimization problems using evolutionary algorithms. In: Proceedings of Fifth International Conference on Multi-Criterion Optimization (EMO-09), pp. 110–124. Heidelberg: Springer (2009)

[14] Deb, K., Sinha, A.: An efficient and accurate solution methodology for bilevel multi-objective programming problems using a hybrid evolutionary-local-search algorithm. Evolutionary Compuation Journal **18**(3), 403–449 (2010)

[15] Dempe, S.: Foundations of bilevel programming. Kluwer, Dordrecht (2002)

[16] Dempe, S., Dutta, J., Lohse, S.: Optimality conditions for bilevel programming problems. Optimization **55**(5-6), 505–524 (2006)

[17] Dempe, S., Dutta, J., Mordukhovich, B.S.: New necessary optimality conditions in optimistic bilevel programming. Optimization **56**(5–6), 577–604 (2007)

[18] Fampa, M., Barroso, L.A., Candal, D., Simonetti, L.: Bilevel optimization applied to strategic pricing in competitive electricity markets. Comput. Optim. Appl. **39**, 121–142 (2008)

[19] Goldberg, D.E., Deb, K., Clark, J.H.: Genetic algorithms, noise, and the sizing of populations. Complex Systems **6**(4), 333–362 (1992)

[20] Goldfarb, D., Idnani, A.: A numerically stable dual method for solving strictly convex quadratic programs. Mathematical Programming **27**, 1–33 (1983)

[21] Hecheng, L., Wang, Y.: A genetic algorithm for solving a special class of nonlinear bilevel programming problems. In: 7th international conference on Computational Science, Part IV: ICCS 2007, pp. 1159–1162 (2007). Also LNCS 4490

[22] Herrmann, J.W.: A genetic algorithm for min-max optimization problems. In: Proceedings of IEEE Congress on Evolutionary Computation (CEC-09), pp. 1099–1103 (2009)

[23] Herskovits, J., Leontiev, A., Dias, G., Santos, G.: Contact shape optimization: A bilevel programming approach. Struct. Multidisc. Optimization **20**, 214–221 (2000)

[24] Jensen, M.T.: A new look at solving minmax problems with coevolution. In: Proceedings of the Fourth Metaheuristics International Conference (MIC-2001), pp. 103–107 (2001)

[25] Malek, A., Yari, A.: Primal-dual solution for the linear programming problems using neural networks. Applied Mathematics and Computation **167**(1), 198–211 (2005)

[26] Miettinen, K., Mäkelä, M.M.: Interactive bundle-based method for nondifferentiable multiobjective optimization: NIMBUS. Optimization **34**, 231–246 (1995)

[27] Mordukhovich, B.S., Nam, N.M.: Subgradients of distance functions at out-of-set points. Taiwanese Journal of Mathematics **10**(2), 299–326 (2006)

[28] Oduguwa, V., Roy, R.: Bi-level optimisation using genetic algorithm. In: Proceedings of the 2002 IEEE International Conference on Artificial Intelligence Systems (ICAIS-02), pp. 322–327 (2002)

[29] Ouorou, A.: A proximal subgradient projection algorithm for linearly constrained strictly convex problems. Optimization Methods Software **22**, 617–636 (2007)

[30] Rao, S.S.: Genetic algorithmic approach for multiobjective optimization of structures. In: Proceedings of the ASME Annual Winter Meeting on Structures and Controls Optimization, vol. 38, pp. 29–38 (1993)

[31] Reklaitis, G.V., Ravindran, A., Ragsdell, K.M.: Engineering Optimization Methods and Applications. New York : Wiley (1983)

[32] Rockafellar, R.T.: Convex Analysis. Princeton University Press (1996)

[33] Sherali, H.D., Choi, G., Ansari, Z.: Limited memory space dilation and reduction algorithms. Comput. Optim. Appl. **19**, 55–77 (2001)

[34] Tulshyan, R., Arora, R., Deb, K., Dutta, J.: Investigating ea solutions for approximate kkt conditions for smooth problems. In: Proceedings of Genetic and Evolutionary Algorithms Conference (GECCO-2010), pp. 689–696 (2010)

[35] Wang, G., Wan, Z., Wang, X., Lv, Y.: Genetic algorithm based on simplex method for solving linear-quadratic bilevel programming problem. Comput. Math. Appl. **56**(10), 2550–2555 (2008)

[36] Zhao, X., Luh, P.B.: New bundle methods for solving lagrangian relaxation dual problems. Journal of Optimization Theory and Applications **113**(2), 373–397 (2002)

[37] Zhou, A., Zhang, Q.: A surrogate assisted evolutionary algorithm for minimax optimization. In: Proceedings of the 2010 Congress on Evolutionary Computation (CEC-10) (2010)

A Dual Solution for Problem P8

Recall that the optimal primal solution to this problem is $\mathbf{x}^P = (110, 10, 110)$ with $f^P = 110$, and the corresponding $\boldsymbol{\lambda}^P = (11, 0.1, 0, 0, 21, 0, 0, 0)$. Both constraints and the variable bound $x_2 \geq 10$ are active at this point. However, here we are interested in the dual solution which corresponds to the maximum of $\theta(\boldsymbol{\lambda})$. The quantity $\theta(\boldsymbol{\lambda})$ itself is defined as the infimum of the Lagrangian function for $\mathbf{x} \in \Re^n$, given below for the problem P8:

$$\begin{aligned}
&L(\mathbf{x}, \boldsymbol{\lambda}) \\
&\quad = \; x_1 + \lambda_1(x_2 + x_3 - 120) + \lambda_2(100x_1 + 100x_2 - x_1x_3 + 100) + \lambda_3(10 - x_1) + \\
&\qquad \lambda_4(x_1 - 10000) + \lambda_5(10 - x_2) + \lambda_6(x_2 - 1000) + \lambda_7(10 - x_3) + \lambda_8(x_3 - 1000), \\
&\quad = \; (1 + 100\lambda_2 - \lambda_3 + \lambda_4)x_1 + (\lambda_1 + 100\lambda_2 - \lambda_5 + \lambda_6)x_2 + (\lambda_1 - \lambda_7 + \lambda_8)x_3 \\
&\qquad - \lambda_2 x_2 x_3 - 120\lambda_1 + 100\lambda_2 + 10\lambda_3 - 10000\lambda_4 + 10\lambda_5 - 1000\lambda_6 + 10\lambda_7 - 1000\lambda_8.
\end{aligned}$$

Since each x_i can take any value on the real line, the quadratic terms in the above expression reaches negative infinity value faster than the linear terms. For this problem, there are two quadratic terms (x_1x_3 and x_2x_3) and thy both appear with λ_2 as a multiplier. Thus, if λ_2 is a non-zero positive value, x_1, x_2 and x_3 can be all increased to either $+\infty$ or all decreased to $-\infty$ to make $L(\mathbf{x}, \boldsymbol{\lambda})$ value reaching its infimum, thereby making $\theta(\boldsymbol{\lambda}) = \infty$. But, since the dual solution corresponds to the maximum value of $\theta(\boldsymbol{\lambda})$, such a $\boldsymbol{\lambda}$-vector (having $\lambda_2 > 0$) cannot be the dual solution. Thus, the dual solution must correspond to $\lambda_2 = 0$.

Substituting $\lambda_2 = 0$ in the above Lagrangian function expression and simplifying, we have the following:

$$\begin{aligned} & L(\mathbf{x}, \boldsymbol{\lambda}) \\ = \;& (1 - \lambda_3 + \lambda_4)x_1 + (\lambda_1 - \lambda_5 + \lambda_6)x_2 + (\lambda_1 - \lambda_7 + \lambda_8)x_3 - 120\lambda_1 + 10\lambda_3 - \\ & 10000\lambda_4 + 10\lambda_5, -1000\lambda_6 + 10\lambda_7 - 1000\lambda_8. \end{aligned}$$

Now, the above expression is linear in terms of $\mathbf{x}$. Hence, if their respective coefficients are non-zero, the corresponding x_i can be monotonically increased to $+\infty$ or decreased to $-\infty$ to make infimum value of $L(\mathbf{x}, \boldsymbol{\lambda})$ to $-\infty$. With a similar argument as above, such a $\boldsymbol{\lambda}$-vector cannot become the dual solution. Hence, all coefficients of x_i terms in the above expression must be uniquely zero. In other words, dual solution ($\boldsymbol{\lambda}^D$) must satisfy the following conditions:

$$\begin{aligned} & 1 - \lambda_3 + \lambda_4 = 0, \\ & \lambda_1 - \lambda_5 + \lambda_6 = 0, \\ & \lambda_1 - \lambda_7 + \lambda_8 = 0. \end{aligned}$$

Replacing λ_3, λ_5 and λ_7 by other λ_i values from the above three expressions, we have the following revised expression:

$$\begin{aligned} & L(\mathbf{x}, \boldsymbol{\lambda}) \\ = \;& -120\lambda_1 + 10\lambda_3 - 10000\lambda_4 + 10\lambda_5 - 1000\lambda_6 + 10\lambda_7 - 1000\lambda_8 \\ = \;& -120\lambda_1 + 10(1 + \lambda_4) - 10000\lambda_4 + 10(\lambda_1 + \lambda_6) - 1000\lambda_6 + 10(\lambda_1 + \lambda_8) - \\ & 1000\lambda_8, \\ = \;& 10 - 100\lambda_1 - 9990\lambda_4 - 990\lambda_6 - 990\lambda_8. \end{aligned}$$

Finally, since the dual solution will correspond to the maximum value of $L(\mathbf{x}, \boldsymbol{\lambda})$, $\lambda_1 = \lambda_4 = \lambda_6 = \lambda_8 = 0$. This causes $\lambda_3 = 1$, $\lambda_5 = 0$ and $\lambda_7 = 0$. Thus, the dual solution is $\boldsymbol{\lambda}^D = (0, 0, 1, 0, 0, 0, 0, 0)$ with $\theta^D = 10$. Clearly, there is a duality gap of $\rho = f^P - \theta^D = 110 - 10 = 10$ for this problem. Interestingly, at this dual solution, $L(\mathbf{x}, \boldsymbol{\lambda}^D)$ expression has no x_i term, thereby making any $\mathbf{x} \in \Re^n$ as a potential solution the Lagrangian subproblem.

B Dual Solution for Problem P9

The primal solution to this problem is $\mathbf{x}^P = (579.3169, 1359.343, 5110.071, 182.0174, 295.5985, 217.9799, 286.4162, 395.5979)$ with $f(\mathbf{x}^P) = 7049.3309$. The corresponding $\boldsymbol{\lambda}^P$-vector for the constraints (without the variable bounds) is (1.9641, 5.2107, 5.1101, 8.476, 9.579, 10), meaning that all constraints are active at the primal solution. However, none of the variable bounds is active at this solution, making $\lambda_i^P = 0$ for $i = 7, \ldots, 22$.

Here, we attempt to derive the dual solution to this problem. The Lagrangian function is defined as follows:

$$\begin{aligned} & L(\mathbf{x}, \boldsymbol{\lambda}) \\ = \;& x_1 + x_2 + x_3 + 1000\lambda_1(0.0025(x_4 + x_6) - 1) + \\ & 1000\lambda_2(0.0025(x_5 + x_7 - x_4) - 1) + 1000\lambda_3(0.01(x_8 - x_5) - 1) + \\ & 0.001\lambda_4(833.33252x_4 + 100x_1 - x_1x_6 - 83333.333) + 0.001\lambda_5(x_2x_4 + 1250x_5 - x_2x_7 - 1250x_4) + \\ & 0.001\lambda_6(x_3x_5 - 2500x_5 - x_3x_8 + 1250000) + \lambda_7(100 - x_1) + \\ & \lambda_8(x_1 - 10000) + \lambda_9(1000 - x_2) + \lambda_{10}(x_2 - 10000) + \\ & \lambda_{11}(1000 - x_3) + \lambda_{12}(x_3 - 10000) + \lambda_{13}(10 - x_4) + \lambda_{14}(x_4 - 1000) + \\ & \lambda_{15}(10 - x_5) + \lambda_{16}(x_5 - 1000) + \lambda_{17}(10 - x_6) + \lambda_{18}(x_6 - 1000) + \\ & \lambda_{19}(10 - x_7) + \lambda_{20}(x_7 - 1000) + \lambda_{21}(10 - x_8) + \lambda_{22}(x_8 - 1000). \end{aligned}$$

Rearranging the terms, we obtain the following:

$$
\begin{aligned}
& L(\mathbf{x}, \boldsymbol{\lambda}) \\
= \; & (1 + 0.1\lambda_4 - \lambda_7 + \lambda_8)x_1 + (1 - \lambda_9 + \lambda_{10})x_2 + (1 - \lambda_{11} + \lambda_{12})x_3 + \\
& (2.5\lambda_1 - 2.5\lambda_2 + 0.83333252\lambda_4 - 1.25\lambda_5 - \lambda_{13} + \lambda_{14})x_4 + \\
& (2.5\lambda_2 - 10\lambda_3 + 1.25\lambda_5 - 2.5\lambda_6 - \lambda_{15} + \lambda_{16})x_5 + (2.5\lambda_1 - \lambda_{17} + \lambda_{18})x_6 + \\
& (2.5\lambda_2 - \lambda_{19} + \lambda_{20})x_7 + (0.01\lambda_3 - \lambda_{21} + \lambda_{22})x_8 - \lambda_4 x_1 x_6 + \lambda_5 x_2 x_4 - \\
& \lambda_6 x_2 x_7 + \lambda_6 x_3 x_5 - \lambda_6 x_3 x_8 - 1000\lambda_1 - 1000\lambda_2 - 1000\lambda_3 - 83.333333\lambda_4 + \\
& 1250\lambda_6 + 100\lambda_7 - 10000\lambda_8 + 1000\lambda_9 - 10000\lambda_{10} + 1000\lambda_{11} - 10000\lambda_{12} + \\
& 10\lambda_{13} - 1000\lambda_{14} + 10\lambda_{15} - 1000\lambda_{16} + 10\lambda_{17} - 1000\lambda_{18} + 10\lambda_{19} - \\
& 1000\lambda_{20} + 10\lambda_{21} - 1000\lambda_{22}.
\end{aligned}
$$

Since all variable bounds are considered as constraints in deriving the Lagrangian function presented above, while finding the infimum value of $L(\mathbf{x}, \boldsymbol{\lambda})$, every possible combination of $x_i \in [-\infty, \infty]$ for $i = 1, \ldots, 5$ must be considered. We notice that there are five quadratic terms involving x_i variables and they involve λ_4, λ_5 and λ_6 as coefficients. A careful thought will reveal that if the coefficients of these five terms are strictly positive, by choosing an appropriate $\mathbf{x}$-vector (such as $\mathbf{x} = (\infty, \infty, \infty, -\infty, -\infty, \infty, \infty, \infty)$), the $L(\mathbf{x}, \boldsymbol{\lambda})$ can be set to its infimum value of $-\infty$. This is because these quadratic terms with x_i values will approach $-\infty$ sooner than the linear terms with x_i alone. Since the dual solution corresponds to the largest value of optimal Lagrangian function value, the coefficients of the quadratic terms must have to be zero for the dual solution. In other words, the above argument reveals that $\lambda_4 = \lambda_5 = \lambda_6 = 0$. Substituting these values to the above expression and simplifying, we obtain the following:

$$
\begin{aligned}
& L(\mathbf{x}, \boldsymbol{\lambda}) \\
= \; & (1 - \lambda_7 + \lambda_8)x_1 + (1 - \lambda_9 + \lambda_{10})x_2 + (1 - \lambda_{11} + \lambda_{12})x_3 + \\
& (2.5\lambda_1 - 2.5\lambda_2 - \lambda_{13} + \lambda_{14})x_4 + (2.5\lambda_2 - 10\lambda_3 - \lambda_{15} + \lambda_{16})x_5 + \\
& (2.5\lambda_1 - \lambda_{17} + \lambda_{18})x_6 + (2.5\lambda_2 - \lambda_{19} + \lambda_{20})x_7 + (10\lambda_3 - \lambda_{21} + \lambda_{22})x_8 - \\
& 1000\lambda_1 - 1000\lambda_2 - 1000\lambda_3 + 100\lambda_7 - 10000\lambda_8 + 1000\lambda_9 - 10000\lambda_{10} + \\
& 1000\lambda_{11} - 10000\lambda_{12} + 10\lambda_{13} - 1000\lambda_{14} + 10\lambda_{15} - 1000\lambda_{16} + 10\lambda_{17} - \\
& 1000\lambda_{18} + 10\lambda_{19} - 1000\lambda_{20} + 10\lambda_{21} - 1000\lambda_{22}.
\end{aligned}
$$

In terms of $\mathbf{x}$-vector, only linear terms remain. Thus, for a given $\boldsymbol{\lambda}$-vector, the coefficients of the linear terms must be zero for the dual solution, thereby outlining the following conditions on the $\boldsymbol{\lambda}$-vector:

$$
\begin{aligned}
& 1 - \lambda_7 + \lambda_8 = 0, \\
& 1 - \lambda_9 + \lambda_{10} = 0, \\
& 1 - \lambda_{11} + \lambda_{12} = 0, \\
& 2.5\lambda_1 - 2.5\lambda_2 - \lambda_{13} + \lambda_{14} = 0, \\
& 2.5\lambda_2 - 10\lambda_3 - \lambda_{15} + \lambda_{16} = 0, \\
& 2.5\lambda_1 - \lambda_{17} + \lambda_{18} = 0, \\
& 2.5\lambda_2 - \lambda_{19} + \lambda_{20} = 0, \\
& 10\lambda_3 - \lambda_{21} + \lambda_{22} = 0.
\end{aligned}
$$

We now replace λ_7, λ_9, λ_{11}, λ_{13}, λ_{15}, λ_{17}, λ_{19} and λ_{21} using other λ_i values by using above conditions and rewrite the Lagrangian function:

$$
\begin{aligned}
& L(x, \lambda) \\
= \; & -1000\lambda_1 - 1000\lambda_2 - 1000\lambda_3 + 100\lambda_7 - 10000\lambda_8 + 1000\lambda_9 - 10000\lambda_{10} + \\
& 1000\lambda_{11} - 10000\lambda_{12} + 10\lambda_{13} - 1000\lambda_{14} + 10\lambda_{15} - 1000\lambda_{16} + 10\lambda_{17} - \\
& 1000\lambda_{18} + 10\lambda_{19} - 1000\lambda_{20} + 10\lambda_{21} - 1000\lambda_{22}, \\
= \; & -1000\lambda_1 - 1000\lambda_2 - 1000\lambda_3 + 100(1 + \lambda_8) - 10000\lambda_8 + 1000(1 + \lambda_{10}) - \\
& 10000\lambda_{10} + 1000(1 + \lambda_{12}) - 10000\lambda_{12} + 10(2.5\lambda_1 - 2.5\lambda_2 + \lambda_{14}) - 1000\lambda_{14} + \\
& 10(2.5\lambda_2 - 10\lambda_3 + \lambda_{16}) - 1000\lambda_{16} + 10(2.5\lambda_1 + \lambda_{18}) - 1000\lambda_{18} + \\
& 10(2.5\lambda_2 + \lambda_{20}) - 1000\lambda_{20} + 10(10\lambda_3 + \lambda_{22}) - 1000\lambda_{22} \\
= \; & -950\lambda_1 - 975\lambda_2 - 1000\lambda_3 - 9900\lambda_8 - 9000\lambda_{10} - 9000\lambda_{12} - 990\lambda_{14} - 990\lambda_{16} - \\
& 990\lambda_{18} - 990\lambda_{20} - 990\lambda_{22} + 2100.
\end{aligned}
$$

For different $\boldsymbol{\lambda}$-vectors, we get different $\theta(\boldsymbol{\lambda})$ values from the above expression. It is clear that the dual solution (maximum $\theta(\boldsymbol{\lambda})$) will take place for zero values for λ_j for $j \in \{1, 2, 3, 8, 10, 12, 14, 16, 18, 20, 22\}$.

The overall dual solution corresponds to $\lambda_7 = \lambda_9 = \lambda_{11} = 1$ and all other $\lambda_j = 0$. The corresponding dual function value is $\theta^D = 2100$. Hence we note a finite duality gap (compared to $f^P = 7049.3309$) in this problem as well. Again, this dual solution corresponds to any $\mathbf{x}$-vector in the n-dimensional real space.