Information Sciences 502 (2019) 537–557

Contents lists available at ScienceDirect

Information Sciences

journal homepage: www.elsevier.com/locate/ins

A multiple surrogate assisted multi/many-objective multi-fidelity evolutionary algorithm

Ahsanul Habib*, Hemant K. Singh, Tapabrata Ray

School of Engineering and Information Technology (SEIT), University of New South Wales (UNSW), Canberra, Australia

ARTICLE INFO

Article history:
Received 2 November 2018
Revised 4 June 2019
Accepted 8 June 2019
Available online 11 June 2019

Keywords:
Multi-fidelity
Multi-objective
Metamodels
Reference directions
Decomposition

ABSTRACT

Engineering design commonly involves optimization of multiple conflicting performance objectives. During the optimization process, the performance of each candidate design/solution is evaluated using a model which may be empirical, numerical, experimental, etc., among other forms. The accuracy of the underlying model in representing the real-world behavior is referred to as *fidelity*. A low-fidelity model may be quick to evaluate but not very accurate; whereas a high-fidelity model may be computationally expensive to evaluate but provides an accurate estimate of the true performance. The paradigm of utilizing the low *and* high-fidelity models' information to identify the high-fidelity optimal solution(s) is known as *multi-fidelity* optimization. This study delves into multi-fidelity optimization for problems which contain multiple objectives and where iterative solvers such as finite element analysis, computational fluid dynamics, etc. are used for performance evaluation. By stopping the solver at various stages *before* convergence, *lower-fidelity* performance estimates can be obtained at reduced computational cost. Most of the existing multi-fidelity methods can only deal with two fidelities (high and low) and a single objective. To overcome this research gap, we present a novel multi-objective evolutionary algorithm that can deal with multiple (arbitrary) number of fidelities by effectively utilizing pre-converged low-fidelity information. The proposed algorithm uses multiple surrogate models to capture the underlying function(s) with enhanced precision. A decomposition-based scheme is deployed for improved scalability in higher number of objectives. A classifier assisted pre-selection method is used to screen potential non-dominated solutions for efficient use of the computational budget. Additionally, a set of multi-fidelity, multi/many objective benchmark problems with different Pareto front types is also introduced to aid a systematic benchmarking. Numerical experiments are presented to highlight the efficacy of the proposed approach.

1. Introduction and background

Design optimization involves evaluation of several candidate designs/solutions in order to find a near optimal or satisfactory solution(s) in a systematic manner. The performance of complex engineering designs is often evaluated through numerical simulations involving iterative solvers such as finite element analysis (FEA), computational fluid dynamics (CFD), computational electromagnetics (CEM), etc. When using sufficiently high-resolution discretization, the solvers are are capa-

* Corresponding author.
E-mail addresses: a.habib@student.adfa.edu.au (A. Habib), h.singh@adfa.edu.au (H. K. Singh), t.ray@adfa.edu.au (T. Ray).

https://doi.org/10.1016/j.ins.2019.06.016
0020-0255/

ble of closely capturing the underlying non-linear physics of complex systems. In contrast, empirical/theoretical equations may (in most cases) require assumptions and simplifications to provide an estimate of performance which might deviate from the system's true behavior. The term "fidelity" is used to refer to how closely a mathematical/numerical model reflects the behavior of a physical system [1]. A high fidelity (HF) model is more accurate but may be costly to evaluate, whereas low fidelity (LF) model may provide coarse estimates quickly.

In some cases, it may be possible that *both* LF and HF models are available for performance evaluation. While the overall aim is to optimize the HF model, the LF models can be used in lieu of the HF models for part of the search in order to save on computational time. An example could be running a CFD analysis to evaluate drag of an airfoil design versus using approximate empirical solutions to Navier–Stokes equation under certain assumptions to estimate the same. Another example could be the use CFD analysis in two different ways - one with a fine mesh (HF) and one with coarse mesh (LF). A third way, which is the key focus of this paper, is to consider pre-converged estimates from the solver as LF and the final converged solution as HF. Thus, the simulation can be terminated a various pre-converged stages to get different fidelities of design performance [2,3].

The use of LF and HF models effectively to achieve this falls under the paradigm of *multi-fidelity* optimization (MFO) [4]. Much of the work done in literature this area deals with problems that contain one objective to be optimized and two available fidelities (one high and one low). However, in practice, engineering design often requires optimization of two or more conflicting performance criteria; the solution to which comprises a set of best trade-off solutions known as Pareto Front (PF). Additionally, it may be possible to have multiple (arbitrary) number of fidelities available; which is particularly true when we consider termination of numerical solutions at various stages. This study is directed towards developing an approach that can handle this challenging class of problems. Additionally, new multi-fidelity multi-objective problems are also proposed to serve as testbed for performance evaluation of the algorithms aimed towards solving such problems.

The remainder of the paper is organized as follows. The related works in the field of computationally expensive and multifidelity optimization are presented in Section 2 in order to establish the research gaps and motivation of this work. The proposed approach is presented in Section 3, while the formulation of the benchmark problems are discussed in Section 4. Numerical experiments are presented in Section 5, followed by conclusions and future research directions in Section 6.

2. Related work

The optimization problems involving computationally expensive evaluations have been traditionally solved using surrogate assisted optimization (SAO) approaches. In SAO, computationally cheap surrogate models (also referred to as approximation models or metamodels) are constructed with the help of the available or systematically sampled high-fidelity evaluations. The surrogates are then used to guide the search in lieu of the true evaluations to save on the computational cost. As more solutions are (selectively) evaluated during the search, the models are updated to improve the estimate of the optimum, and the process is repeated until the stopping criterion is met. Development of SAO techniques is an active area of research delving into various types of surrogates and model management strategies. The interested readers may refer to [5] for a general overview of SAO methods. As evident, the input to the conventional SAO methods is only one (typically high) fidelity of the given design problem. Multi-fidelity optimization (MFO) is relevant when multiple different models are available to evaluate the performance measures. Selected representative studies in MFO are discussed below.

2.1. Single-objective multi-fidelity optimization

Traditionally, MFO problems have been studied within the paradigm of single-objective optimization. Some notable methods employed in single-objective MFO include the use of scaling functions between LF and HF evaluations [6], hierarchical/multilevel methods [7], dimensionality reduction techniques [8] and space mapping techniques [9] among others. There have been studies to use the surrogate modeling within MFO as well. Some studies along this line focused on predicting the scaling function with the help of Gaussian process/Kriging [10], response surface models (RSM) [11], radial basis functions (RBF) [12] and support vector machines (SVM) [13]. Some other ways in which surrogates have been used within MFO include combining the LF and HF information to construct co-Kriging models [14], using HF variables in addition to the LF variables for constructing local and global surrogates [15] and multi-step adaptive sampling using Kriging and RBF models [16].

Most single-objective MFO methods can handle only 2 fidelity levels (an LF and an HF), and very few studies have explored strategies to handle more than this. Among these, both SAO and non-SAO evolutionary algorithm (EA) exist. One of the recent study along non-SAO methods is [3] which focused on progressively switching fidelity levels via rank-reversal prediction models. Another recent study [1] proposed a generation based, an individual based and a hybrid fidelity adjustment strategy along with a generic single objective, unconstrained, multi-fidelity test suite for MFO benchmarking. Some SAO-based techniques for handling any number of fidelity models include expected improvement (EI) based sampling strategies [17], correlation based evaluation techniques [18] and use of co-Kriging [19]. Most of these studies focused on progressive evaluation of fidelity models which could potentially devote a disproportionate computational effort searching the lower fidelities that may be far from HF optimum. To alleviate this shortcoming, the authors have previously proposed a multiple surrogate assisted single-objective multi-fidelity EA [2]. The key enhancement was the ability to evaluate any solution using any (as opposed to progressive) LF model based on the local correlation between LF with the HF models.

2.2. Multi-objective multi-fidelity optimization

Although the development of single-objective MFO methods has attracted significant research attention, a number of real-world problems involve several conflicting objectives which need to be optimized simultaneously. Optimization problems with 2 or more objectives are referred to as multi-objective problems (MOPs). MOPs with 4 or more objectives are further sub-categorized as many-objective problems (MaOPs) due to additional challenges they introduce to optimization methods, decision making and visualization [20]. The optimum to MOP comprises a set of trade-off solutions referred to as Pareto Set (PS) in the decision space, and its image PF in the objective space. Multi-objective evolutionary algorithms (MOEAs) are commonly employed for solving MOPs due to their capability of tackling problems that are highly non-linear, non-differentiable, discontinuous and inherently *black-box* in nature. However, since MOEAs evolve a *population* of solutions over a number of generation to obtain a PF approximation, they are not suitable for solving computationally expensive optimization problems in their basic form. Therefore, surrogate models are often employed in conjunction with MOEAs to obtain a good PF approximation with low evaluation budget. Some prominent works along this area include the use of EI maximization based efficient global optimization (EGO) [21,22] for solving MOPs (extendable to MaOPs). Moreover, a few recent studies have been proposed which are designed specifically for solving expensive MaOPs, such as a classification-based approach using a feed-forward neural network (CSEA) [23] and Kriging-based reference vector guided EA, K-RVEA [24]. A recent review of the surrogate assisted EAs for solving expensive MOPs/MaOPs can be found in [25].

Despite an increasing number of works being reported in handling expensive MOPs/MaOPs, multi-fidelity problems and approaches have received scarce attention in this context. Most of the studies related to this topic only focused on MOPs with only 2 or 3 fidelity levels. Some initial studies [26,27] used gradient-based weighted-sum methods for finding a few solutions along some pre-defined reference vectors on the PF. However, the use of gradient based solvers would face significant challenges in solving problems with local optima (e.g. DTLZ1, DTLZ3 [28]). In [29], a multi-level MOEA was designed for solving a turbine blade aerodynamic optimization problem, while [30] implemented a Kriging [31] assisted MOEA for designing a planar Yagi antenna. Later, Zhu et. al. [32] extended the study presented in [6] to multi-objective cases within MOEA framework with the help of Kriging and proposed a novel fixed update strategy for surrogate refinement. Recently, Zhou et. al. [33] proposed a Kriging assisted multi-fidelity EA which enhanced the performance of the method presented in [32]. In this study, they first obtained a preliminary PF by running the LF model with the help of NSGA-II [34] and evaluated selected solutions based on crowding distance measure in HF. Thereafter, they constructed a multi-fidelity surrogate model by tuning the LF model with the help of Kriging prediction of the scaling function. They also proposed strategies to update the Kriging model after each generation and after a pre-specified number of generations.

In this paper, we attempt to address some of the gaps identified in the discussion above. More specifically, the contributions include:

- Simultaneously handling a multiple (arbitrary number of) fidelity levels and multiple objectives. The problems involving iterative solvers fit this description well since the simulation can be stopped/started at to obtain solutions of different levels of fidelity.
- Further, we also propose a set of benchmark test problems based on the renowned DTLZ [28] scalable test suite and adapting the error functions from [1] to mathematically represent different fidelity levels to test the proposed algorithm on problems with different PF shapes.
- Extensive numerical experiments are presented to demonstrate the efficacy of the proposed approach over other existing strategies.

3. Proposed approach

An unconstrained multi-fidelity multi/many-objective optimization problem can be mathematically represented as shown in Equation 1.

$$\begin{aligned} &\underset{(\mathbf{x})}{\text{Minimize}}\ f_1(x, L), f_2(x, L), \ldots f_M(x, L) \\ &\qquad x \in \mathbb{R}^D, \quad x_{lb} \leq x \leq x_{ub} \end{aligned} \tag{1}$$

Here, the problem contains D decision variables with lower bounds, $\mathbf{x}_{lb}$ and upper bound $\mathbf{x}_{ub}$ and M is the number of objectives. L denotes the computation of objective function f in the highest level of fidelity (HF). The objective function can also be computed in lower levels of fidelity i.e. for $l = 1, 2, 3, \ldots, L-1$. For example, if the problem has 7 fidelity levels, $l = 7$ will represent the HF model, whereas $l = 1, 2, \ldots, 6$ will represent the (increasingly accurate) LF models.

To solve this problem efficiently, we propose an algorithm we refer to as M3EA – which stands for **Multiple surrogate-assisted**, **Multi/many-objective**, **Multi-fidelity** EA. The prominent factor considered in designing M3EA is the assumption that each function evaluation is computationally expensive which requires a significantly "long" time (e.g. a few hours/days) and some coarse estimate (with varying accuracy) of the actual function is available as LF models. As the problem is considered computationally expensive, the algorithmic "overheads" such as construction/training of surrogates, recombination or selection operators, finding nearest neighboring solutions, computing local correlation coefficient (Kendall's τ), updating archive etc. are considered negligible compared to an actual expensive function evaluation.

Hence, the components of M3EA are designed to employ a number of mechanisms (discussed below) to improve the likelihood that the solution(s) that is/are selected for evaluation in the next fidelity level is/are promising solutions i.e. most likely to be ND solutions. M3EA inherits some of the basic concepts from existing literature, such as reference vector (RV) adaptation strategy [35] and use of multiple surrogate models [2]. However, [2] can deal with only single-objective problems, while [35] is not design to handle multiple fidelity levels or computationally expensive problems. Thus, the new notable features of M3EA are as follows:

- It can deal with problems having multiple fidelity levels and multiple objectives.
- It employs a novel cost saving strategy where for a promising solution only the objective with the lowest local correlation with HF is evaluated instead of evaluating all objectives in the next fidelity level.
- A two-stage filtering is proposed for identifying potential HF ND solutions. First, a classifier based pre-selection of promising solutions is used, followed by an RV based selection of promising solutions.

The pseudo-code of the approach is presented in Algorithm 1, while the details of the key individual components are discussed in following subsections.

3.1. Generation of reference vectors and initialization of population

A set of RVs, W_0 is generated using the systematic sampling on unit simplex using the method proposed in [36], with the origin at the ideal point. The ideal point is constructed using the best objective values of the population in each objective, denoted as $z_{\min}$. The objective space is normalized based on the minimum and maximum objective values of the population. Hence, in the normalized objective space, $z_{\min} = 0^M$. The set W_0 consists of N_W points on the hyperplane with a uniform spacing $d = 1/H$ for any number of objectives M with H unique sampling locations along each objective axis. For problems with more than 6 objectives, a 2-layered approach is followed for W_0 as proposed in [37]. An initial population of N ($=N_W$) candidate solution vectors is generated using Latin Hypercube Sampling (LHS).

3.2. Archive update

The N initial solutions are evaluated up to the HF level. Since we consider iterative solvers, an evaluation in fidelity level L would imply that we have its performance available at all fidelity levels 1 through to L. All initial solutions are stored in an archive $\mathcal{A}$. Further, the ND solutions in $\mathcal{A}$ are identified and stored in another archive $\mathcal{A}_{ND}$. If any solution during the course of search is evaluated in the HF level L, the information is updated in both the archives.

3.3. Offspring generation

In this study, we employ Simulated Binary Crossover (SBX) [34] and Differential Evolution (DE) [38] *DE/current-to-rand* 2 followed by Polynomial Mutation (PM) [34] to generate an offspring solution. As the SBX operator generates two offspring solutions, one of them is chosen randomly. The offspring solutions are evaluated in the first fidelity level in all objectives. However, whether or not the offspring solution is evaluated in next fidelity level depends upon its performance in the environmental selection process (discussed shortly).

3.4. Construction of multiple surrogate models and prediction of the HF estimate

Multiple surrogate models are constructed for each objective function in a similar to our previous study on the single objective multi-fidelity problems [2]. Among the available surrogate modeles, we have used response RSM [39] with polynomial degree of 1 and 2 (termed as RSM1 and RSM2) and RBF [40] in this study. The first two models are more suited to linear/quadratic regression, whereas the last one is suited to more generic non-linear response functions. Using them together offers a flexibility to chose the model most suited to a given (unknown) function according to their performance (validation error) and consequently result in more accurate performance predictions.

In the first generation, upon evaluating a promising offspring solution in the first fidelity level ($l = 1$), the HF estimate for a particular objective function is predicted using the best local surrogate model built using neighboring solutions from archive $\mathcal{A}$. The input of the surrogate models are the N_{is} neighboring designs from $\mathcal{A}$ and the corresponding first fidelity function values (denoted as x_{nn} and $f^{fid}_{nn_j}$ in Algorithm 1), while the output is the corresponding HF values (indicated as $f^{L}_{nn_j}$ in Algorithm 1). In this study, $\min(\lfloor 6D \rfloor, |\mathcal{A}|)$ neighbors from $\mathcal{A}$ are chosen as the neighborhood. Before going into the training phase, the inputs are normalized. The x_{nn} are normalized based on their upper and lower bounds and f^{fid}_{nn} is normalized based on the lowest and highest values of the chosen fidelity level, *fid* in the archive. The input is first divided into training and validation sets. In this study, 80% of the input points are used for training and the remaining 20% is used for validation (similar to [2]). For a given solution, the surrogate model with the minimum Root Mean-Squared Error (*RMSE*) on the validation set is chosen to predict its HF performance.

Algorithm 1 M3EA.

Input: D = Number of decision variables, N_W = No. of RVs, W_0 = Set of RVs in normalized hyperplane, N = Population size (=N_W), N_{init} = Initial population size (11D-1), L = Number of fidelity levels, N_{ns} = Number of neighboring solutions, S = Surrogate models used, $C_{j,l}$ = Evaluation cost in fidelity levels $l = 1, 2, \ldots, L$ for objectives $j = 1, 2, \ldots, M$, $cost_{\max}$ = Maximum computational budget.

```
cost = 0
W_0 = InitializeRVs(N_W, M)
Initialize (P)                                   ▷ Population of N solutions
Evaluate^{1:L}_{1:M}(P)                          ▷ Evaluate initial population, P in all fidelity levels for all objectives
A = UpdateArchive(P)
A_ND = UpdateNDArchive(A)                        ▷ Find ND solutions in the archive and store them separately
cost = UpdateCost(P, C)
while (cost < cost_max) do
    O = GenerateOffsprings(N, P)                 ▷ Generate N unique offspring solutions from P
    for i = 1 : |O| do
        for j = 1 : M do
            fid = FindFidelity(O_i, j)           ▷ Current fidelity level O_i is evaluated in for jth objective
            if fid = 0 then
                Evaluate^{fid+1}_j(O_i)          ▷ Evaluate O_i in the first fidelity level of the jth objective if it has not been evaluated before
            end if
            (x_nn, f^{1:L}_{nn_j}) = FindNearestNeighbors(O_i, N_ns, A)   ▷ Find N_ns nearest neighboring solutions of O_i from the archive A
            if fid < L then
                x_in = [x_nn, f^{fid}_{nn_j}]    ▷ Use the x_nn and the f^{fid}_{nn_j} as training input to surrogate models
            else
                x_in = x_nn                      ▷ Use only the x_nn as training input to surrogate models
            end if
            y_in = f^L_{nn_j}
            (x_train, y_train, x_valid., y_valid.) = Select training and validation points from x_in and y_in
            l = 1                                ▷ Initialization for the best surrogate model
            for k = 1 : |S| do
                Model_k = TrainModel(S_k, x_train, y_train)
                RMSE_k = CalculateRMSE(Model_k, x_valid., y_valid.)
                if k > 1 then
                    if RMSE_k < RMSE_{k-1} then
                        l = k                    ▷ Best surrogate model so far
                    else
                        l = k-1
                    end if
                end if
            end for
            Predict^L_j(O_i, Model_l)            ▷ Predict O_i's highest fidelity response in jth objective
        end for
    end for
    P = P+O
    (z_min, z_max) = FindMinMax(P)               ▷ Find the minimum and maximum objective values of the current population P
    W = Adapt(W_0, z_min, z_max)                 ▷ Adapt W_0 using z_min and z_max resulting in W
    Assign P to W based on the smallest acute angle
    T = ∅
    while W ≠ ∅ do
        Identify non-empty RVs, W_NE.            ▷ RVs with at-least one soln. assigned to them
        for w = 1 : |W_NE| do
            R_w = Solns. assigned to W_{NE_w}.
            d_w = Euclidean distances from solns. in R_w to z_min
            I = Soln. with minimum Euclidean distance    ▷ With smallest value in d_w
            I = UpdateFidelityLevel(I)
            T = T+I
        end for
        W = W-W_NE
    end while
    P = T
    A = UpdateArchive(P)
    A_ND = UpdateNDArchive(A)
    cost = UpdateCost(P, C)
end while
Return A_ND.
```

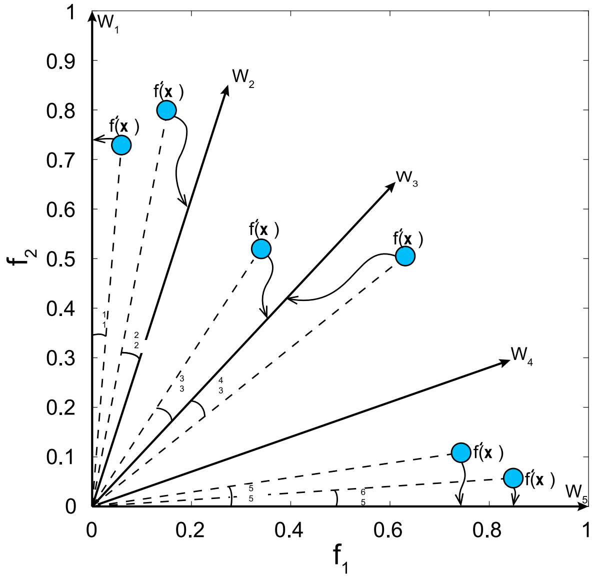

Figure 1. Assignment of solutions to RVs.

3.5. Environmental selection

The environmental selection process consists of the steps detailed in the following sub-subsections:

3.5.1. Adaptation of reference vectors

After predicting the HF estimates, the parent and the offspring solutions are combined with their respective actual and predicted HF values. Then, the minimum and maximum values, i.e., $z_{\min}$ and $z_{\max}$, of the combined population is calculated and the adaptation of the RVs is as per Eqn. (2) (proposed in [35]).

$$W_i = \frac{W_{0,i} \odot (z_{\max} - z_{\min})}{||W_{0,i} \odot (z_{\max} - z_{\min})||}; \quad i = 1, \ldots, N_W, \tag{2}$$

In the above equation, W_i and $W_{0,i}$ are the ith adapted and initially generated RVs, $(z_{\max} - z_{\min})$ is the difference between the best and worst objective values of the current population, and $\odot$ is the Hadamard product for element-wise multiplication of two vectors of equal size.

3.5.2. Assignment

For assigning the solutions of the combined parent+offspring population to W, the objective values of the solutions are translated i.e., $f'_j(x_i) = f_j(x_i) - z_{\min_j}$, where, $f_j(x_i)$ is the value of the jth objective of the ith solution in the population and $z_{\min_j}$ is the minimum value of the jth objective in the current population. After that, the acute angles between a solution and all RVs are calculated [35]. A solution is assigned to an RV which has the smallest acute angle with that solution. This process divides the population into different sub-populations. The process is illustrated in Figure 1 for a 2-objective case for 5 RVs and 6 solutions. The RVs are indicated as $W_1, \ldots, W_5$ and the translated objective vectors of the solutions are presented as $f'(x_1), \ldots, f'(x_6)$. The acute angle between a solution and its closest RV (in the objective space having the minimum acute angle between the solution and the RV) is denoted as θ^a_b, where a is the solution no. and b is the no. of RV. As, solution no. a is making the smallest acute angle with RV no. b and hence, solution a is assigned to RV no. b. As seen from Figure 1, the RVs W_1 and W_2 each are assigned one solution each (solutions x_1 and x_2) according to θ^1_1 and θ^2_2. On the other hand, W_3 and W_5 both are assigned with 2 solutions each. Solutions x_3 and x_4 are assigned to W_3 with smallest acute angles θ^3_3 and θ^4_3 while solutions x_5 and x_6 are assigned to W_5 according to smallest acute angles θ^5_5 and θ^6_5. One can notice that W_4 does not have any solutions assigned to it and thus remains empty.

3.5.3. Selection of the best solution along each reference vector

After the above assignment, the environmental selection process identifies solutions which are to be carried forward to the next generation as parents. For the non-empty RVs, one solution is selected from each of them based on the smallest Euclidean distance (ED) from the $z_{\min}$. Next, the non-empty RVs are removed and the combined set of solutions are assigned to the currently empty RVs. The same process of selection is followed iteratively to select a total of N solutions as parents. Figure 2 illustrates a scenario with 8 RVs and 12 solutions. Initially, W_3 and W_6 are empty and solutions S_2, S_5, S_6, S_8, S_{10}

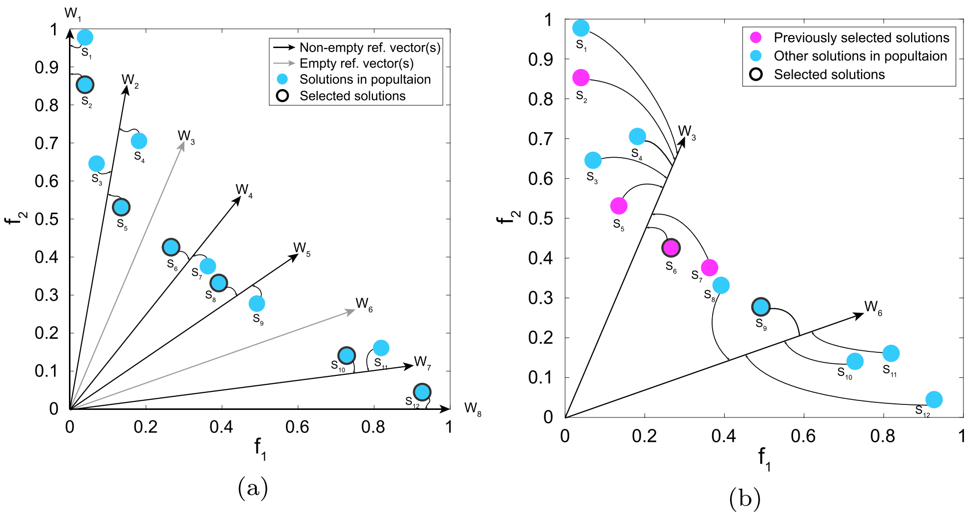

Figure 2. Environmental selection process (a) selecting solutions from the non-empty RVs (b) selecting solutions from the initial empty RVs.

and S_{12} are selected along W_1, W_2, W_4, W_5, W_7 and W_8. Next, all other non-empty RVs are removed and only W_3 and W_6 are retained. All solutions are assigned again and the solutions with minimum ED are chosen along these RVs. Please note that the solution S_6 is selected twice in this process and we create a copy of S_6 in the selected set of solutions to accommodate this.

3.5.4. Fidelity update

Fidelity update is the process of evaluating a promising solution (selected in the previous step) in a higher fidelity level. For a given solution, the currently evaluated fidelity is first identified for each objective. One can then evaluate this solution in the next fidelity level for all objectives. However, considering a very low computational budget, one can also opt for saving some computational cost by evaluating the solution in the next fidelity level for only selected objectives (instead of all). In this study, we have implemented both strategies (with and without saving cost) in our proposed algorithm and investigated the respective performances.

In the algorithm variant with cost saving option enabled, the objective to be evaluated in the next fidelity level is identified by observing the local correlation measure of the solution's current fidelity level with the HF level for all objectives. The objective with the lowest correlation measure is chosen for fidelity update. The Kendall's rank correlation coefficient, τ is used in this study (similar to our previous study [2]) as the correlation measure. The local τ values are computed by identifying the nearest neighbors of the solution from the archive $\mathcal{A}$. The number of nearest neighbors and their selection process is the same as in Section 3.4.

3.6. Evaluation cost

The initial population undergoes full evaluation (i.e. they are evaluated in all fidelity levels for all objectives). This results in the initialization cost, denoted as $cost_{init}$. The promising offspring solutions are initially evaluated in the first fidelity level and later can be evaluated in next fidelity levels based on the procedures discussed above. The cost incurred this way is the evaluation cost, denoted as $cost_{eval}$. Using the LF estimates of the offspring solutions, one can predict their HF performance. At any given generation where the current performance of the algorithm is to be observed, the population needs to undergo a full evaluation to obtain the HF values. The cost associated with this process is referred as the fill-up cost denoted as $cost_{fillup}$. The total initial cost is the summation of $cost_{init}$ for all solutions in the parent population. On the other hand, after each generation, the total cost is calculated by adding the $cost_{eval}$ for all solutions, adding the $cost_{fillup}$ of the solutions in the current population and subtracting the $cost_{fillup}$ of the previous generation. The $cost_{fillup}$ is calculated after every generation to keep track of the progress of the search with regards to the HF in the event the search is stopped in the current generation. If the search undergoes another generation, the $cost_{fillup}$ of the previous generation is subtracted and this way, we only truly incur the $cost_{fillup}$ for the generation in which the evolutionary search terminates.

3.7. Illustrative example

In this sub-section, we present an example to illustrate the component used for saving evaluation cost. To show this, we have chosen the MFMaOP1 problem (formulation is discussed in next section) with the instability error, which randomly adds a large number with a particular fidelity level following to a pre-defined probability. In Figure 3 we compare the

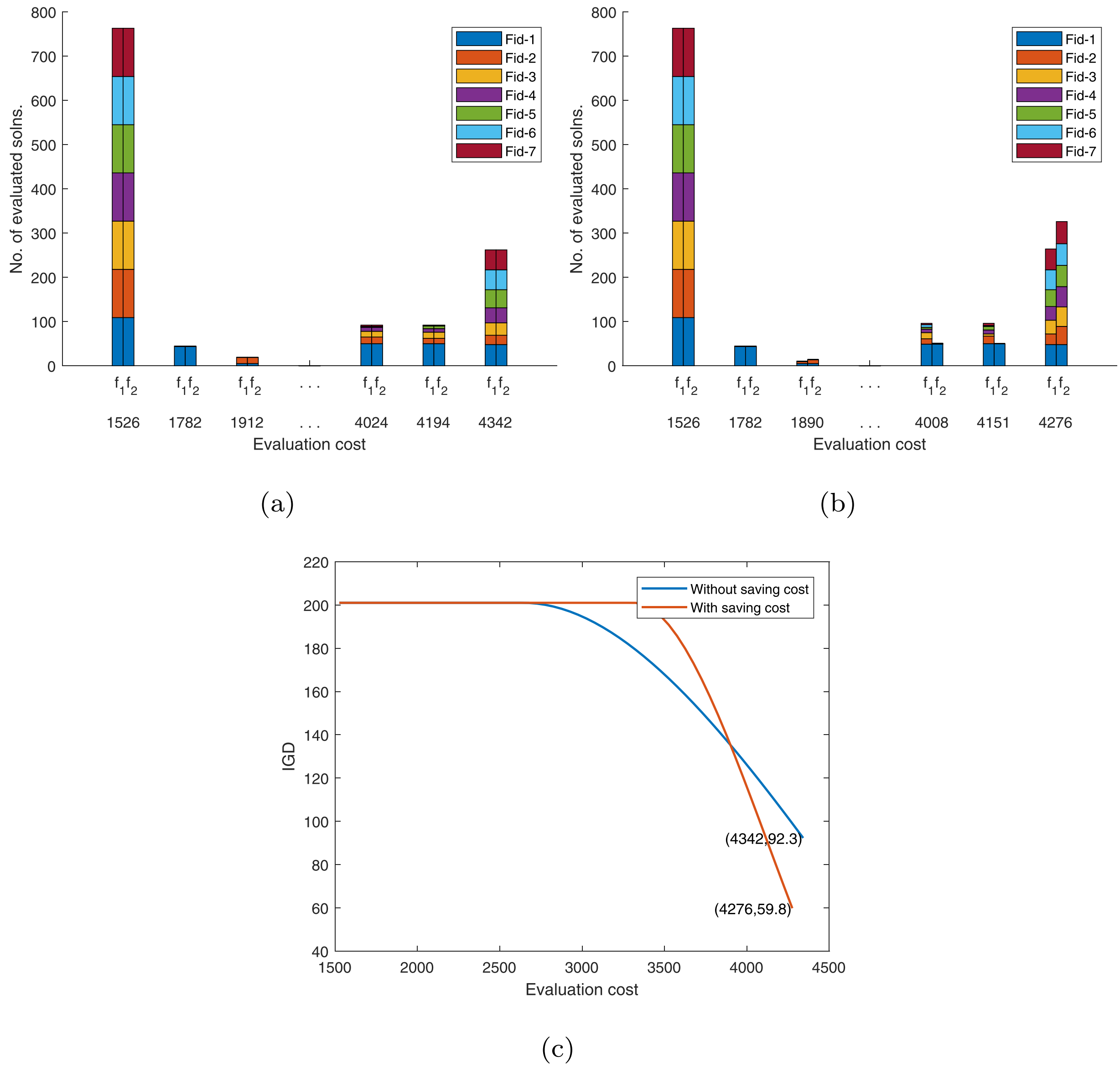

Figure 3. Number of evaluated solutions in different fidelity levels for objectives f_1 and f_2 for MFMaOP1 problem with instability error type 1: (a) Without saving cost (b) With saving cost (c) Comparison of IGD convergence after 17 generations.

results of the baseline M3EA without and with cost saving option enabled. The figure presents the different number of function evaluations for each fidelity level for each option. Also shown is the convergence plot of the Inverted Generational Distance (IGD) metric over the evaluation cost.

The problem is run for a fixed number of generations (17 for this example) to observe the benefit of enabling the saving evaluation cost option. From Figure 3(a) and (b), firstly, we can observe the number of evaluation cost after 17 generations for both options. The baseline M3EA without cost saving option uses 4342 units of cost while the version with cost saving option enabled uses 4276 units. Now, if we look at the IGD convergence plot of both variants in Figure 3(c), the benefit of cost saving would becomes clearer. The M3EA variant with cost saving not only requires less evaluation cost, but also delivers better result in terms of IGD value. One might argue about the slow converging nature of the variant with cost saving option during the initial part of the run. However, this behavior is expected as it evaluates only the first objective in different fidelity levels as it has a poor minimum correlation among its different LF levels and the HF level. For this reason, the algorithm needs to rely on the surrogate models for the highest fidelity estimates for the second objective. As we update the archive only when a solution is evaluated in the HF level for all objectives, the archive update process slows down for the variant when we opt for cost saving which affects the IGD. However, as it saves some evaluation costs, it gets more opportunity to explore the search space (if the surrogate model does not misguide the search). Although the variant without cost saving initially performs well, it spends the evaluation cost for both objectives from the beginning and hence delivers a relatively sub-par result. Please take note that if the correlation between various fidelity levels with the highest fidelity level is equally poor for all objectives, both variants are expected to perform equivalently. An equivalent performance is also expected if the evaluation budget is increased and the surrogate models are trained with a higher number of points and hence are able to more closely capture the underlying objective function landscapes.

4. Formulation of the benchmark problems

In order to study the performance of algorithms aimed towards solving multi-fidelity multi/many-objective optimization problems, we systematically construct and study a set of benchmark problems that capture characteristic challenges involved. We have designed the benchmark problems incorporating error functions proposed in [1] into the well-known DTLZ multi-objective test problem suite [28]. The chosen problems, DTLZ1, DTLZ2, DTLZ2^{-}, DTLZ5 and DTLZ7, represent diverse types of PF shapes, i.e., linear, non-convex, convex, degenerated and disconnected, respectively. Moreover, the error functions have been adopted to emulate three different types of errors, i.e., resolution, stochastic and instability, which are commonly encountered in MFO [1].

For an M-objective, D-dimensional multi-objective multi-fidelity problem, if $f_j(x)$ indicates the HF objective function of its jth objective, the expression of a fidelity level l can be written as:

$$f_j(x, l) = f_j(x) + e(x, l), \tag{3}$$

where $x \in \mathbb{R}^D$; $j = 1, 2, \dots, M$ and $l = 1, 2, \dots, L$. The function $e(x, l)$ introduces a certain type of error to the HF function that varies with the different LF levels. In this study, the value of l ranges from 1 to 7, where, 1 indicates the lowest fidelity level and 7 represents the highest fidelity level or the true function. The evaluation cost associated with different fidelity levels (for each objective) is assumed to be $C_{j,l} = l$ for simplicity in this study. However, other types of cost functions can be easily accommodated in the algorithm.

In [1], four types of resolution and stochastic errors (i.e. $e_{res}^{1\text{ to }4}$ and $e_{stoc}^{1\text{ to }4}$, respectively) and 2 types of instability errors ($e_{ins}^{1\&2}$) were proposed. In this study, we have incorporated e_{res}^3, e_{stoc}^3, e_{ins}^1 and e_{ins}^2 into the selected DTLZ problems to obtain the multi-fidelity benchmark problems. For simplicity of representation, we refer to e_{res}^3, e_{stoc}^3, e_{ins}^1 and e_{ins}^2 as E_{res}, E_{stoc}, E_{ins}^1 and E_{ins}^2 throughout the paper. Their mathematical forms are shown in Equations 4–7.

$$\begin{aligned}
&E_{res}(x, l) = \sum_{i=1}^{D} \alpha(l)\cos(\omega(l)x_i + \beta(l) + \pi) \\
&\textit{where}, \\
&\alpha(l) = \begin{cases} 0.8000 & \text{if, } l = 1 \\ 0.7000 & \text{if, } l = 2 \\ 0.6000 & \text{if, } l = 3 \\ 0.4000 & \text{if, } l = 4 \\ 0.2000 & \text{if, } l = 5 \\ 0.1000 & \text{if, } l = 6 \\ 0 & \text{otherwise} \end{cases}; \\
&\omega(l) = 10\pi\alpha(l); \\
&\beta(l) = 0.5\pi\alpha(l)
\end{aligned} \tag{4}$$

$$\begin{aligned}
&E_{stoc}(x, l) = N(\mu(x, l), \sigma(l)) \\
&\textit{where}, \\
&\mu(x, l) = 0.1\nu(l)\gamma(x)/D; \\
&\sigma(l) = 0.1\nu(l); \\
&\nu(l) = \begin{cases} 0.9000 & \text{if, } l = 1 \\ 0.7500 & \text{if, } l = 2 \\ 0.6000 & \text{if, } l = 3 \\ 0.4500 & \text{if, } l = 4 \\ 0.3000 & \text{if, } l = 5 \\ 0.1500 & \text{if, } l = 6 \\ 0 & \text{otherwise} \end{cases};
\end{aligned} \tag{5}$$

Here, N is a Gaussian distribution with a mean of μ and a standard deviation of σ.

$$\begin{aligned}
&E^1_{ins}(x,l) = \begin{cases} \phi^1 & \text{if, } r \le pr^1(l) \\ 0 & \text{if, } r > pr^1(l) \end{cases}; \\
&where, \\
&\phi^1 = 10 \times |x|; \\
&pr^1(l) = \begin{cases} 0.0900 & \text{if, } l = 1 \\ 0.0750 & \text{if, } l = 2 \\ 0.0600 & \text{if, } l = 3 \\ 0.0450 & \text{if, } l = 4 \\ 0.0300 & \text{if, } l = 5 \\ 0.0150 & \text{if, } l = 6 \\ 0 & \text{otherwise} \end{cases};
\end{aligned} \tag{6}$$

$$\begin{aligned}
&E^2_{ins}(x,l) = \begin{cases} \phi^2 & \text{if, } r \le pr^2(l) \\ 0 & \text{if, } r > pr^2(l) \end{cases}; \\
&where, \\
&\phi^2 = 10 \times |x|; \\
&pr^2(l) = \begin{cases} 0.3329 & \text{if, } l = 1 \\ 0.0743 & \text{if, } l = 2 \\ 0.0166 & \text{if, } l = 3 \\ 0.0037 & \text{if, } l = 4 \\ 0.0008 & \text{if, } l = 5 \\ 0.0002 & \text{if, } l = 6 \\ 0 & \text{otherwise} \end{cases};
\end{aligned} \tag{7}$$

Here, ϕ is a large number, r is a random number in [0,1] and $pr(l)$ is the probability of generating an outlier in the simulation. The generic formulation of an M objective problem represented as shown in Equation 8.

$$\begin{aligned}
&\text{Minimize} \\
&f_1(x,l) && = \text{DTLZ}\chi_{f_1} \times E(x,l), \\
&f_2(x,l) && = \text{DTLZ}\chi_{f_2} \times E(x,l), \\
&\vdots \\
&f_{M-1}(x,l) && = \text{DTLZ}\chi_{f_{M-1}} \times E(x,l), \\
&f_M(x,l) && = \text{DTLZ}\chi_{f_M} \times E(x,l), \\
&where, \\
&0 \le x_i \le 1; && \text{for } i = 1, 2, \ldots, D, \\
&\chi \in \{1,\ 2,\ 2^-,\ 5,\ 7\}, \\
&E \in \{E_{res},\ E_{stoc},\ E^1_{ins},\ E^2_{ins}\}.
\end{aligned} \tag{8}$$

1) MFMaOP1: The problem is constructed based on the DTLZ1 problem. The highest fidelity is the original DTLZ1 problem which has a linear PF. The Pareto-optimal solutions for the highest fidelity function correspond to $x^*_M = 0$. The PF for an M-objective problem lies on the linear hyperplane with $\sum_{i=1}^{M} f_i = 0.5,\ f_i > 0$ for $i = 1, \ldots, M$ for the highest fidelity.
2) MFMaOP2: The problem is formulated based on the DTLZ2 problem. The highest fidelity is the original DTLZ2 problem which has a non-convex PF. The Pareto-optimal solutions for the highest fidelity correspond to $x^*_M = 0.5$ and the PF comprises the orthant where $\sum_{i=1}^{M} f_i^2 = 1,\ f_i > 0$ for $i = 1, \ldots, M$.
3) MFMaOP2⁻: The problem is designed based on the highest fidelity being the DTLZ2⁻ problem proposed in [41]. This variant considers maximization of DTLZ2 objectives, or equivalently, minimization of negated objective of original DTLZ2. The resultant PF is an inverted form of the PF of original DTLZ2, and is convex in shape.
4) MFMaOP3: The problem is developed from the DTLZ5 problem. The highest fidelity is the DTLZ5 problem which has a degenerate PF. The Pareto-optimal solutions for the highest fidelity correspond to $x^*_M = 0.5$ and the PF corresponds to the set of solutions with $\sum_{i=1}^{M} f_i^2 = 1,\ f_i > 0$ for $i = 1, \ldots, M$.
5) MFMaOP4: This problem is constructed considering the original DTLZ7 problem as the highest fidelity. It has a disconnected PF shape comprising four distinct patches. The PF corresponds to the solutions with $x^*_M = 0$.

5. Numerical experiments

In this section, we quantitatively assess the performance of M3EA. Towards this end we consider the problems dicussed in the previous section with 2, 3, 6, 8 and 10 objectives. The evaluation budgets are set to 4200, 6300, 12600, 16,800 and 21,000 units for 2, 3, 6, 8 and 10 objectives, which is equivalent to conducting 300 HF evaluations in each objective. The

Table 1
Spacing parameters H_1 and H_2 and the corresponding number of reference vectors N_W for different numbers of objectives M.

M	2	3	6	8	10
(H_1, H_2)	(49,0)	(13,0)	(4,1)	(3,2)	(3,2)
N_W	50	105	132	156	275

Table 2
Number of points in reference sets for IGD calculation for different values of M.

	No. of points in reference sets	
M	MFMaOP4	Other problems
2	3000	3000
3	6084	5050
6	59,049	33,649
8	78,125	50,338
10	19,683	92,378

number of variables is set to 10 for all. The number of RVs for different objectives along with the spacing values H_1 and H_2 are presented in Table 1.

The performance of M3EA is compared with the following strategies:

1. Progressive fidelity optimization: In this strategy, the total cost is distributed equally between all fidelity levels. Each fidelity is used progressively, starting from the lowest to the highest, for evaluating the objective functions during the search. We refer to this algorithm as *PrF*, which is representative of the existing strategies that use an increasingly accurate function evaluation over the course of optimization [42].
2. Individual fidelities: In this strategy, a chosen individual fidelity level is used for the entire course of optimization. Correspondingly, the algorithms are simply named by the chosen fidelity level utilized for the optimization, i.e., *Fid-1, Fid-2, . . . Fid-7*.

5.1. Performance metrics

The IGD [43] metric is used for quantifying the quality of the PF approximations obtained in the experiments. The IGD is able to provide a composite measure of convergence and diversity, which makes it one of the commonly used metrics in benchmarking MOEAs. To compute IGD, a reference set needs to be specified that is a close representation of the target PF (known for benchmark problems). The reference sets have been generated using the PlatEMO [44] framework. For the *MFMaOP2* $^-$ problem, the reference sets are derived from the DTLZ2 problem as suggested in [41]. The number of points in the reference sets for different objectives are listed in Table 2.

To check the statistical significance of the results obtained by different algorithms over multiple instantiations, the Wilcoxon's Rank-sum (WRS) [45] test is performed on the median IGD values with a 5% confidence level and 25 independent runs of the compared algorithms are performed for each problem considered.

Further, to assess the cumulative performance of different algorithms across the problems, the performance profile plots [46] are used. Performance profile is a statistical tool for observing how well or how fast and what proportion of problem instances were solved by a particular algorithm relative to the best performing algorithm in each problem instance. The horizontal axis of the performance-profile plot shows the goal value τ (in this case, the ratio of the median best IGD of an algorithm compared to the best performing algorithm in a particular problem instance) and the vertical axis represents the cumulative distribution ($\rho_s(\tau)$) of the median best IGD value. The $\rho_s(\tau)$ is the indication of the proportion of problem instances an algorithm is capable of solving within a factor τ with respect to the best performing algorithm. Thus, different algorithms can be compared on a given goal value τ.

5.2. Effectiveness of the 'cost saving' component

In the first set of experiments, we investigate the effectiveness of enabling the cost saving option. For this, we run the experiments on the benchmark problems with and without the cost saving option enabled in M3EA. The average IGD values obtained over 25 independent runs for all benchmark problems for all error instances are presented in Table 3 along with the statistical significance test (WRS test) results. More detailed statistics (minimum, average and maximum) is presented in Table IV of the supplementary material. We also list the total numbers of problem instances (n), wins (w), losses (l) and ties (t) of the M3EA variant with cost saving option disabled (labeled as M3EA (SaveCostOff)) when compared with the variant with cost saving option enabled (labeled as M3EA (SaveCostOn)).

Table 3
Test results for mean IGD metric obtained by M3EA (SaveCostOn) and M3EA (SaveCostOff). The best mean results are highlighted in bold. Here, ↓, ↑ and ≈ indicates whether M3EA (SaveCostOff) is statistically significantly better, worse or equivalent to M3EA (SaveCostOn).

Prob.	Error Type	M	M3EA (SaveCostOn)		M3EA (SaveCostOff)
MFMaOP1	Resolution	2	72.7354	≈	**72.2679**
		3	**56.7942**	↑	61.5434
		6	**23.0774**	≈	25.3240
		8	9.4851	≈	**9.2562**
		10	**0.5404**	≈	0.5404
	Stochastic	2	**66.6029**	↑	78.8427
		3	**58.9604**	≈	60.6402
		6	**22.2276**	≈	23.9926
		8	**7.8852**	≈	8.4484
		10	**0.4965**	≈	0.4975
	Instability-1	2	**74.4468**	≈	78.9558
		3	**63.7239**	≈	68.0974
		6	27.8898	≈	**26.1561**
		8	11.3031	≈	**9.9101**
		10	0.5398	≈	**0.5352**
	Instability-2	2	**83.8415**	↑	95.4580
		3	77.2374	≈	**77.1622**
		6	**26.4237**	≈	27.7730
		8	**9.3225**	≈	9.7775
		10	0.5071	≈	**0.5062**
MFMaOP2	Resolution	2	0.2957	≈	**0.2611**
		3	**0.3346**	≈	0.3715
		6	**0.5396**	↑	0.5749
		8	**0.6304**	≈	0.6404
		10	0.6690	≈	**0.6690**
	Stochastic	2	0.2588	≈	**0.2274**
		3	**0.3263**	↑	0.3701
		6	**0.5076**	↑	0.5457
		8	**0.5888**	≈	0.6052
		10	0.6346	≈	**0.6339**
	Instability-1	2	0.2887	≈	**0.2572**
		3	**0.4153**	≈	0.4158
		6	**0.6311**	≈	0.6379
		8	0.6368	≈	**0.6305**
		10	0.6452	≈	**0.6431**
	Instability-2	2	**0.2006**	≈	0.2511
		3	**0.3247**	↑	0.3646
		6	**0.5180**	↑	0.5590
		8	**0.5853**	≈	0.6013
		10	0.6372	≈	**0.6366**
MFMaOP2	Resolution	2	**0.1760**	≈	0.1761
		3	0.4401	≈	**0.4384**
		6	1.1146	≈	**1.1128**
		8	**1.6450**	≈	**1.6450**
		10	**2.3431**	≈	2.3431
	Stochastic	2	4.1099	≈	**4.0704**
		3	**3.9566**	≈	3.9837
		6	**3.8010**	↑	3.8330
		8	**3.7179**	↑	3.7279
		10	3.6765	≈	**3.6763**
	Instability-1	2	4.1098	≈	**4.0768**
		3	**3.9763**	≈	3.9954
		6	3.8446	≈	**3.8426**
		8	**3.7299**	≈	3.7304
		10	**3.6750**	≈	3.6752
	Instability-2	2	**4.0579**	≈	4.0939
		3	**3.9416**	↑	3.9852
		6	**3.8210**	≈	3.8317
		8	**3.7167**	↑	3.7312
		10	3.6762	≈	**3.6747**

(*continued on next page*)

Table 3 (*continued*)

Prob.	Error Type	M	M3EA		M3EA
MFMaOP3	Resolution	2	0.2614	≈	**0.2589**
		3	0.2341	≈	**0.2259**
		6	0.1293	≈	**0.1290**
		8	0.0658	≈	**0.0651**
		10	**0.0140**	≈	0.0140
	Stochastic	2	**0.1483**	≈	0.1568
		3	**0.1707**	≈	0.1744
		6	**0.1070**	≈	0.1076
		8	**0.0600**	≈	0.0611
		10	**0.0141**	≈	0.0143
	Instability-1	2	0.1672	≈	**0.1647**
		3	**0.1811**	≈	0.1849
		6	0.1139	≈	**0.1125**
		8	0.0606	≈	**0.0590**
		10	**0.0142**	≈	0.0143
	Instability-2	2	**0.1191**	≈	0.1269
		3	**0.1642**	≈	0.1696
		6	0.1037	≈	**0.1006**
		8	**0.0553**	≈	0.0553
		10	**0.0139**	≈	0.0141
MFMaOP4	Resolution	2	1.5849	≈	**1.3016**
		3	**2.5962**	≈	2.7196
		6	**3.3932**	≈	3.9935
		8	**3.5756**	≈	3.9109
		10	1.2148	≈	**1.2111**
	Stochastic	2	1.4592	≈	**1.3493**
		3	**2.4887**	≈	2.8090
		6	**3.5045**	↑	4.8888
		8	4.1187	≈	**3.6986**
		10	1.1883	≈	**1.1873**
	Instability-1	2	2.0545	≈	**1.5812**
		3	**3.0317**	≈	3.1593
		6	**5.1094**	≈	5.4657
		8	4.1355	≈	**3.6738**
		10	1.1810	≈	**1.1747**
	Instability-2	2	1.3170	≈	**1.2284**
		3	**2.7861**	≈	2.9120
		6	**4.2636**	≈	4.4418
		8	3.3170	≈	**3.2014**
		10	1.1830	≈	**1.1750**
	n/w/l/t		–/–/–		100/13/0/87

Observing the 'wins' (better performance) and 'losses' (worse performance) according to the WRS test from Table 3, we can observe that the two variants mostly performs equivalently; however, the variant with cost saving option enabled delivering slightly better results in some problems. The M3EA variant with cost saving option scores 13 wins and suffers no losses in the 100 problem instances. On the other hand, both variants deliver statistically equivalent results in terms of IGD on 87 problem instances. However, if we look at the performance profile plots (obtained from the median IGD among 25 independent runs) of both variants in Figure 4, we can observe that up to almost 60% cases both variants exhibit similar performances (i.e they were best or equal best). However, for the remainder of the ≈40% instances, it can be seen that the performance of M3EA (SaveCostOn) deviated much less from the best value compared to M3EA (SaveCostOff).

Hence, for offering greater reliability in case of unknown problems, we propose to select the cost saving option. As discussed previously, in the worst case scenario, if the correlation between the different fidelity levels and the highest fidelity level is equally poor for all objectives, the two variants will deliver almost similar results. On the flip-side, if the cost saving option is enabled, the algorithm will have more opportunity to explore the search space compared to the variant with no cost saving option and hence, may deliver better results if the surrogate models provide reasonable estimates of the highest fidelity functions. Moreover, if some objectives have reasonable correlation for its lower fidelities (with the highest fidelity function), the algorithm will take advantage of it and will only evaluate the solutions in the objective that has the worst correlation. This will not only save evaluation cost but also guide the search better towards the Pareto optimal solutions.

5.3. Results on the benchmark problems

Having presented the potential benefits of the cost saving option, we now benchmark the performance of M3EA with the other strategies discussed earlier, namely *PrF* and *Fid-1* to *Fid-7*. The compared algorithms are implemented within the same

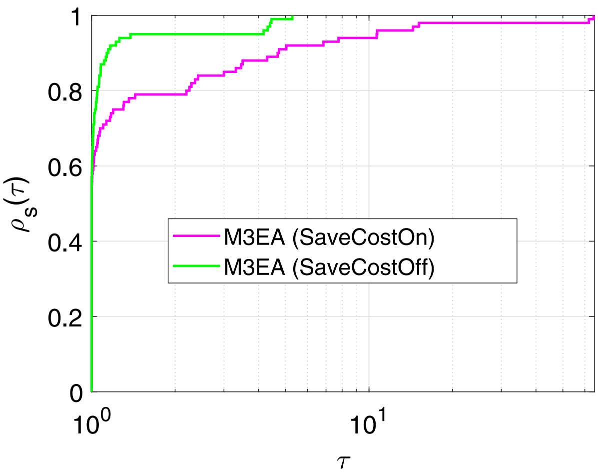

Figure 4. Performance profile plots of the two M3EA variants with and without cost saving option enabled considering the median IGD values of 25 independent runs.

framework for the sake of a fair comparison. That is, they evaluate the initial population in the highest fidelity, predict the highest fidelity estimates with the help of multiple surrogate models by constructing local surrogate models with the help of the neighboring solutions in the same fidelity level. Moreover, they update the archive and the surrogate models with the solutions that are best along each RV after the environmental selection. Finally, when the search is terminated, the final population is evaluated in the highest fidelity level which reflects the $cost_{fillup}$ of the proposed M3EA.

The average IGD obtained by the algorithms from 25 independent runs is presented in Table 4 while the detailed statistics indicating the minimum, average and maximum IGD along with the WRS test results are presented in Tables V and VI of the supplementary file. The WRS test results for individual problem instances are presented along with the total numbers of wins, losses and ties for each algorithm in the above-mentioned tables. The results clearly indicate that the proposed M3EA shows significantly better results compared to progressively optimizing the highest fidelity level as well as using individual fidelity levels. When the *PrF* is compared with M3EA, the proposed M3EA scores 91 wins, concedes 2 losses and delivers statistically equivalent results in 7 problem instances. On the other hand, when *Fid-4, 5* and *6* are compared with M3EA, they perform similarly with M3EA winning in 95 instances and delivering equivalent results in 5 instances with no loses in any instance. A similar behavior is observed in case of *Fid-1* to *Fid-3* when compared to M3EA. Among these three algorithms, *Fid-3* performs comparatively worse as M3EA wins in 96 instances, losing in 1 instance and delivering equivalent results in 3 instances. The *Fid-2* algorithm again performs slightly better than *Fid-3* algorithm when compared to M3EA as the later wins in 92 instances, loses in none and performs equivalent in 8 problem instances. Finally, when the *Fid-1* is compared against M3EA, the proposed algorithm wins in 86 instances, loses in 3 instances and ties in 11 instances. Finally, the best performing algorithm among all these fixed fidelity algorithms is the *Fid-7* i.e. optimization with the help of the highest fidelity level. According to the statistical significance test (WRS test), M3EA scores win in 75 instances, suffers loss in 9 instances while in the rest of the instances (16) they deliver equivalent results.

The performance of the compared algorithms can be further visually observed from the performance profile plot (obtained from median IGD values of 25 independent runs) in Figure 5. The figure indicates a clear performance advantage of M3EA over other algorithms. The median IGD values obtained using M3EA are observed to be within a factor of 1.8 from the best performing algorithm for any problem instance, while this factor is more than 5 for the other algorithms. Furthermore, the PF approximations obtained from different algorithms under study are presented in Figures 1–6 of Section III of the supplementary file for some selected problems for the readers to visually observe the distribution of the obtained solutions.

5.4. Further enhancements through pre-selection of promising offspring solutions

In this sub-section we propose a pre-selection mechanism to further enhance the performance of M3EA. This pre-selection technique is incorporated to identify promising solutions from the offspring and discard the rest before spending any evaluation cost on them. The "promising" solutions refer to those which are highly likely to be ND solutions. For pre-selecting the likely ND solutions, we need a classifier which is able to predict the classes of the solutions as ND or not ND. In this study, we have employed the widely used SVM classifier [47]. With the help of the archive of evaluated solutions with all fidelity information, $\mathcal{A}$ and the archive of all ND solutions according to the highest fidelity level, $\mathcal{A}_{ND}$ a classifier model is built to predict the class of the offspring solutions.

The classifier model is constructed with the decision variables as inputs and the archive solution labels as outputs. The label of an ND solution is *true* (1) and the label of any other solution is *false* (0). The implementation of SVM available in MATLAB®'s two-class built-in function `fitcsvm` is used with all options set to its default. Whenever an offspring solution

Table 4
Test results for mean IGD metric obtained by M3EA, PrF and Fid-1 to 7. The best results are highlighted in bold. Here, ↓, ↑ and ≈ indicates whether the particular algorithm is statistically significantly better, worse or equivalent to M3EA.

Prob.	Error Type	M	M3EA		PF		Fid-1		Fid-2		Fid-3		Fid-4		Fid-5		Fid-6		Fid-7
MFMaOP1	Resolution	2	76.9945	↑	117.0938	≈	**70.7296**	↑	92.0772	↑	94.0301	↑	98.9147	↑	112.1348	↑	112.9670	↑	111.8331
		3	**56.7942**	↑	98.0348	↑	109.5452	↑	108.3619	↑	107.2731	↑	107.2731	↑	108.2381	↑	108.2381	↑	87.6344
		6	**23.0774**	↑	35.3039	↑	40.6934	↑	40.6879	↑	39.9414	↑	40.0306	↑	39.4242	↑	39.4489	↑	30.8173
		8	**9.4851**	↑	13.1010	↑	15.0118	↑	15.2367	↑	14.9958	↑	15.3224	↑	14.9958	↑	14.9958	≈	10.9491
		10	0.5404	≈	0.6098	≈	0.6294	≈	0.6294	≈	0.6294	≈	0.6294	≈	0.6294	≈	0.6294	≈	**0.4899**
	Stochastic	2	68.0806	↑	109.5008	↓	**65.8343**	≈	85.2110	↑	101.9279	↑	93.8685	↑	108.6675	↑	119.8102	↑	117.6693
		3	**58.9604**	↑	105.7556	↑	119.4974	↑	119.4974	↑	119.4974	↑	119.4974	↑	119.4974	↑	119.4974	↑	84.1267
		6	**22.2276**	↑	37.1182	↑	43.1816	↑	43.1816	↑	43.1816	↑	43.1816	↑	43.1816	↑	43.1816	↑	32.7864
		8	**7.8852**	↑	12.0965	↑	15.0512	↑	15.0509	↑	15.0512	↑	15.0512	↑	15.0512	↑	15.0512	≈	10.0846
		10	**0.4965**	≈	0.5688	↑	0.6294	↑	0.6294	↑	0.6294	↑	0.6294	↑	0.6294	↑	0.6294	≈	0.5107
	Instability-1	2	**77.4676**	↑	116.6200	≈	80.3618	↑	99.1560	↑	103.8383	↑	107.1770	↑	118.0687	↑	115.6432	↑	117.0363
		3	**63.7239**	↑	101.7782	↑	114.1304	↑	114.6074	↑	114.1985	↑	114.3916	↑	114.1304	↑	114.1304	↑	94.7499
		6	**27.8898**	↑	36.6665	↑	43.7264	↑	43.6021	↑	43.0984	↑	44.3207	↑	42.0844	↑	43.9696	≈	30.7187
		8	11.3031	≈	12.8579	≈	13.7325	≈	13.8353	↑	14.8979	≈	13.2069	≈	13.2490	≈	13.1373	≈	**10.2266**
		10	0.5398	≈	0.5766	≈	0.6294	≈	0.6294	≈	0.6294	≈	0.6294	≈	0.6294	≈	0.6294	≈	**0.5005**
	Instability-2	2	87.3475	↑	114.5472	≈	**83.0828**	↑	97.0946	↑	108.8766	↑	107.3287	↑	114.5642	↑	116.5224	↑	117.0363
		3	**77.2374**	↑	99.7572	↑	115.7757	↑	114.6074	↑	114.1304	↑	114.1304	↑	114.1304	↑	114.1304	↑	94.7499
		6	**26.4237**	↑	38.0907	↑	43.7330	↑	43.6021	↑	43.2846	↑	42.1808	↑	41.9477	↑	44.0408	↑	30.7187
		8	**9.3225**	≈	11.5771	↑	15.7057	↑	13.8353	↑	13.2487	↑	13.1373	↑	13.2490	↑	13.1373	≈	10.3202
		10	0.5071	≈	0.5766	↑	0.6294	↑	0.6294	↑	0.6294	↑	0.6294	↑	0.6294	↑	0.6294	≈	**0.5005**
MFMaOP2	Resolution	2	**0.3064**	↑	0.9161	↑	0.9465	↑	0.9294	↑	0.9965	↑	0.9337	↑	0.8584	↑	0.7986	↑	0.8247
		3	**0.3346**	↑	1.1092	↑	1.1604	↑	1.1351	↑	1.1589	↑	1.1310	↑	1.1230	↑	1.1119	↑	0.9326
		6	**0.5396**	↑	0.8615	↑	0.9049	↑	0.8973	↑	0.9025	↑	0.8978	↑	0.9027	↑	0.8837	↑	0.7660
		8	**0.6304**	↑	0.7756	↑	0.8054	↑	0.8049	↑	0.8023	↑	0.8093	↑	0.8062	↑	0.7955	↑	0.7085
		10	**0.6690**	↑	0.7384	↑	0.7545	↑	0.7545	↑	0.7545	↑	0.7545	↑	0.7545	↑	0.7545	↑	0.7024
	Stochastic	2	**0.2654**	↑	0.7993	↑	0.3163	↑	0.5356	↑	0.6817	↑	0.7029	↑	0.8159	↑	0.8109	↑	0.8212
		3	**0.3263**	↑	1.0499	↑	1.1344	↑	1.1373	↑	1.1397	↑	1.1362	↑	1.1369	↑	1.1327	↑	0.9383
		6	**0.5076**	↑	0.8214	↑	0.8972	↑	0.8966	↑	0.8970	↑	0.8925	↑	0.8950	↑	0.8944	↑	0.7646
		8	**0.5888**	↑	0.7653	↑	0.8036	↑	0.8034	↑	0.8062	↑	0.8034	↑	0.8018	↑	0.8010	↑	0.7026
		10	**0.6346**	↑	0.7340	↑	0.7545	↑	0.7545	↑	0.7545	↑	0.7545	↑	0.7545	↑	0.7545	↑	0.6988
	Instability-1	2	0.8149	↑	0.7761	≈	**0.3632**	↑	0.6376	↑	0.7152	↑	0.7532	↑	0.8526	↑	0.7934	↑	0.8204
		3	**0.8969**	↑	1.0433	↑	1.1385	↑	1.1339	↑	1.1324	↑	1.1294	↑	1.1324	↑	1.1335	↑	0.9397
		6	0.9350	↑	0.8429	↑	0.9012	↑	0.8908	↑	0.8909	↑	0.8919	↑	0.8896	↑	0.8912	↑	**0.7479**
		8	0.8693	↑	0.7600	↑	0.8061	↑	0.8047	↑	0.7962	↑	0.7965	↑	0.8008	↑	0.7976	↑	**0.7065**
		10	0.8357	↑	0.7345	↑	0.7545	↑	0.7545	↑	0.7545	↑	0.7545	↑	0.7545	↑	0.7545	↑	**0.6994**
	Instability-2	2	**0.2041**	↑	0.7535	↑	0.6878	↑	0.6434	↑	0.6770	↑	0.7696	↑	0.8440	↑	0.8564	↑	0.8185
		3	**0.3247**	↑	1.0219	↑	1.1597	↑	1.1339	↑	1.1350	↑	1.1254	↑	1.1317	↑	1.1324	↑	0.9397
		6	**0.5181**	↑	0.8440	↑	0.9098	↑	0.8908	↑	0.8959	↑	0.8895	↑	0.8878	↑	0.8878	↑	0.7479
		8	**0.5853**	↑	0.7610	↑	0.8075	↑	0.8047	↑	0.7962	↑	0.7986	↑	0.7960	↑	0.7968	↑	0.7087
		10	**0.6372**	↑	0.7345	↑	0.7545	↑	0.7546	↑	0.7545	↑	0.7545	↑	0.7546	↑	0.7551	↑	0.6994
MFMaOP2	Resolution	2	0.1760	↓	0.1667	↓	0.1647	≈	0.1734	↓	0.1625	≈	0.1665	≈	0.1734	≈	0.1761	↓	**0.1398**
		3	0.4401	↓	0.3950	≈	0.4324	↑	0.4354	↑	0.4348	≈	0.4356	≈	0.4325	↑	0.4411	↓	**0.3461**
		6	**1.1146**	↑	1.3240	↑	1.3893	↑	1.3873	↑	1.3896	↑	1.3864	↑	1.3917	↑	1.3810	↑	1.1812
		8	**1.6450**	↑	1.9349	↑	2.0078	↑	2.0097	↑	2.0085	↑	2.0062	↑	2.0064	↑	2.0017	↑	1.7926
		10	**2.3431**	↑	2.5150	↑	2.5531	↑	2.5531	↑	2.5531	↑	2.5531	↑	2.5531	↑	2.5531	↑	2.4473
	Stochastic	2	**4.1102**	↑	4.5516	↑	4.1310	↑	4.3119	↑	4.4317	↑	4.4569	↑	4.5907	↑	4.5586	↑	4.6081
		3	**3.9566**	↑	4.5918	↑	4.6868	↑	4.6914	↑	4.6946	↑	4.6904	↑	4.6926	↑	4.6877	↑	4.4635
		6	**3.8010**	↑	4.0101	↑	4.0758	↑	4.0745	↑	4.0734	↑	4.0706	↑	4.0721	↑	4.0698	↑	3.9618
		8	**3.7179**	↑	3.8081	↑	3.8438	↑	3.8440	↑	3.8429	↑	3.8439	↑	3.8414	↑	3.8411	↑	3.7725
		10	**3.6765**	↑	3.7173	↑	3.7311	↑	3.7311	↑	3.7311	↑	3.7311	↑	3.7311	↑	3.7311	↑	3.6966

(*continued on next page*)

Table 4 (*continued*)

Prob.	Error Type	M	M3EA		PF		Fid-1		Fid-2		Fid-3		Fid-4		Fid-5		Fid-6		Fid-7
	Instability-1	2	**0.8149**	↑	4.5517	↑	4.2049	↑	4.3906	↑	4.4788	↑	4.5113	↑	4.5622	↑	4.5679	↑	4.5833
		3	**0.8969**	↑	4.5780	↑	4.6806	↑	4.6700	↑	4.6700	↑	4.6633	↑	4.6748	↑	4.6746	↑	4.4784
		6	**0.9350**	↑	4.0234	↑	4.0813	↑	4.0753	↑	4.0713	↑	4.0798	↑	4.0709	↑	4.0722	↑	3.9745
		8	**0.8693**	↑	3.8061	↑	3.8515	↑	3.8471	↑	3.8459	↑	3.8439	↑	3.8441	↑	3.8428	↑	3.7740
		10	**0.8357**	↑	3.7151	↑	3.7311	↑	3.7311	↑	3.7311	↑	3.7311	↑	3.7311	↑	3.7311	↑	3.6966
	Instability-2	2	**4.0592**	↑	4.5226	↑	4.4721	↑	4.4002	↑	4.4702	↑	4.5477	↑	4.5727	↑	4.6050	↑	4.5829
		3	**3.9416**	↑	4.5708	↑	4.6979	↑	4.6700	↑	4.6743	↑	4.6586	↑	4.6625	↑	4.6699	↑	4.4784
		6	**3.8211**	↑	4.0094	↑	4.0890	↑	4.0753	↑	4.0750	↑	4.0727	↑	4.0711	↑	4.0703	↑	3.9745
		8	**3.7167**	↑	3.8124	↑	3.8551	↑	3.8472	↑	3.8459	↑	3.8417	↑	3.8426	↑	3.8404	↑	3.7751
		10	**3.6760**	↑	3.7151	↑	3.7314	↑	3.7311	↑	3.7311	↑	3.7311	↑	3.7311	↑	3.7311	↑	3.6966
MFMaOP3	Resolution	2	0.2714	↑	0.3199	↑	0.3204	↑	0.3162	↑	0.3366	↑	0.3108	↑	0.3587	↑	0.2962	↓	**0.2399**
		3	0.2341	↑	0.2848	↑	0.3038	↑	0.3017	↑	0.3040	↑	0.3018	↑	0.3054	↑	0.2976	↓	**0.2318**
		6	0.1293	↑	0.1553	↑	0.1609	↑	0.1603	↑	0.1600	↑	0.1598	↑	0.1626	↑	0.1609	↓	**0.1205**
		8	0.0658	↑	0.0769	↑	0.0823	↑	0.0813	↑	0.0813	↑	0.0817	↑	0.0830	↑	0.0805	↓	**0.0588**
		10	0.0140	↑	0.0188	↑	0.0218	↑	0.0218	↑	0.0218	↑	0.0218	↑	0.0218	↑	0.0218	↓	**0.0138**
	Stochastic	2	0.1542	↑	0.2297	↑	**0.1330**	↑	0.1859	↑	0.2065	↑	0.2199	↑	0.2503	↑	0.2450	↑	0.2486
		3	**0.1707**	↑	0.2618	↑	0.2917	↑	0.2900	↑	0.2890	↑	0.2892	↑	0.2889	↑	0.2891	↑	0.2350
		6	**0.1070**	↑	0.1428	↑	0.1572	↑	0.1563	↑	0.1558	↑	0.1565	↑	0.1562	↑	0.1547	↑	0.1244
		8	**0.0600**	↑	0.0708	↑	0.0816	↑	0.0810	↑	0.0805	↑	0.0803	↑	0.0800	↑	0.0797	≈	0.0604
		10	0.0141	↑	0.0182	↑	0.0218	↑	0.0218	↑	0.0218	↑	0.0218	↑	0.0218	↑	0.0218	↓	**0.0139**
	Instability-1	2	0.7191	↑	0.2324	≈	**0.1513**	↑	0.2014	↑	0.2144	↑	0.2235	↑	0.2523	↑	0.2575	↑	0.2408
		3	0.7897	↑	0.2628	↑	0.2901	↑	0.2907	↑	0.2926	↑	0.2899	↑	0.2901	↑	0.2930	↑	**0.2297**
		6	0.8595	↑	0.1463	↑	0.1580	↑	0.1557	↑	0.1548	↑	0.1539	↑	0.1544	↑	0.1569	≈	**0.1219**
		8	0.8733	↑	0.0708	↑	0.0792	↑	0.0798	↑	0.0773	↑	0.0783	↑	0.0778	↑	0.0794	≈	**0.0597**
		10	0.8846	↑	0.0190	↑	0.0218	↑	0.0218	↑	0.0218	↑	0.0218	↑	0.0218	↑	0.0218	↓	**0.0134**
	Instability-2	2	**0.1233**	≈	0.2260	≈	0.2395	≈	0.1992	≈	0.2293	↑	0.2526	↑	0.2519	↑	0.2507	≈	0.2410
		3	0.3789	↑	0.2679	↑	0.2992	↑	0.2907	↑	0.2912	↑	0.2949	↑	0.2896	↑	0.2901	↑	**0.2297**
		6	0.6777	↑	0.1467	↑	0.1598	↑	0.1557	↑	0.1572	↑	0.1568	↑	0.1552	↑	0.1551	↑	**0.1219**
		8	0.8157	↑	0.0746	↑	0.0792	↑	0.0797	↑	0.0796	↑	0.0786	↑	0.0781	↑	0.0777	↑	**0.0606**
		10	0.9073	↑	0.0190	↑	0.0218	↑	0.0218	↑	0.0218	↑	0.0218	↑	0.0218	↑	0.0218	≈	**0.0134**
MFMaOP4	Resolution	2	**1.6301**	↑	3.2672	≈	1.6885	≈	2.4909	↑	2.9225	↑	2.9513	↑	3.3259	↑	3.4855	↑	3.1033
		3	**2.6392**	↑	5.9383	↑	6.4843	↑	6.4522	↑	6.4522	↑	6.4522	↑	6.4825	↑	6.4451	↑	5.5393
		6	**3.3392**	↑	9.1957	↑	9.8010	↑	9.8193	↑	9.8010	↑	9.8010	↑	9.8010	↑	9.8010	↑	7.1040
		8	**3.9832**	↑	6.2413	↑	8.0938	↑	7.4725	↑	7.4725	↑	7.4732	↑	7.4728	↑	7.4725	↑	5.3734
		10	**1.2107**	↑	1.5116	↑	1.7016	↑	1.7016	↑	1.7016	↑	1.7016	↑	1.7016	↑	1.7016	↑	1.2720
	Stochastic	2	**1.4504**	↑	3.3496	↓	1.7276	≈	2.5129	↑	2.8544	↑	3.1485	↑	3.4403	↑	3.4601	↑	3.4632
		3	**2.5822**	↑	5.8550	↑	6.4722	↑	6.4535	↑	6.4559	↑	6.4559	↑	6.4559	↑	6.5144	↑	5.5775
		6	**4.2733**	↑	8.4855	↑	9.5578	↑	9.5681	↑	9.5578	↑	9.5578	↑	9.5578	↑	9.5578	↑	7.0922
		8	**3.9558**	↑	6.0132	↑	7.4994	↑	7.1604	↑	7.1604	↑	7.1604	↑	7.1604	↑	7.1604	≈	4.2759
		10	**1.1849**	↑	1.5351	↑	1.7016	↑	1.7016	↑	1.7016	↑	1.7016	↑	1.7016	↑	1.7016	↑	1.2525
	Instability-1	2	4.9371	↑	3.8365	↑	3.9774	↑	4.1665	↑	4.3274	↑	4.5432	↑	4.0028	↑	3.8945	↑	**3.3358**
		3	7.4300	↑	6.2938	↑	6.7406	↑	6.8059	↑	6.6556	↑	6.7695	↑	6.7948	↑	6.6459	↑	**5.5184**
		6	12.6974	↑	8.2459	↑	9.7093	↑	9.4791	↑	9.7391	↑	9.5071	↑	9.4289	↑	9.4553	↑	**7.4747**
		8	14.4796	↑	6.7282	↑	8.1060	↑	8.0045	↑	7.8215	↑	7.4414	↑	7.3057	↑	7.6389	≈	**5.1911**
		10	12.4007	↑	1.4860	↑	1.7016	↑	1.7016	↑	1.7016	↑	1.7016	↑	1.7016	↑	1.7016	↑	**1.2911**
	Instability-2	2	3.8295	↑	3.4243	↑	4.1999	↑	4.2457	↑	3.4431	↑	**3.2477**	↑	3.3673	↑	3.2785	↑	3.3179
		3	6.7174	↑	6.0536	↑	6.6804	↑	6.8059	↑	6.5870	↑	6.5743	↑	6.5849	↑	6.5825	↑	**5.5208**
		6	12.8704	↑	7.1715	↑	9.7865	↑	9.4791	↑	9.5427	↑	9.4850	↑	9.5266	↑	9.4289	↑	**7.1597**
		8	13.9431	↑	6.7660	↑	7.5719	↑	8.0045	↑	7.3975	↑	7.5029	↑	7.8781	↑	7.4462	↑	**5.0482**
		10	12.2166	↑	1.4860	↑	1.7016	↑	1.7016	↑	1.7016	↑	1.7016	↑	1.7016	↑	1.7016	↑	**1.2911**
	n/w/l/t	–/–/–			100/91/2/7		100/86/3/11		100/92/0/8		100/96/1/3		100/95/0/5		100/95/0/5		100/95/0/5		100/75/9/16

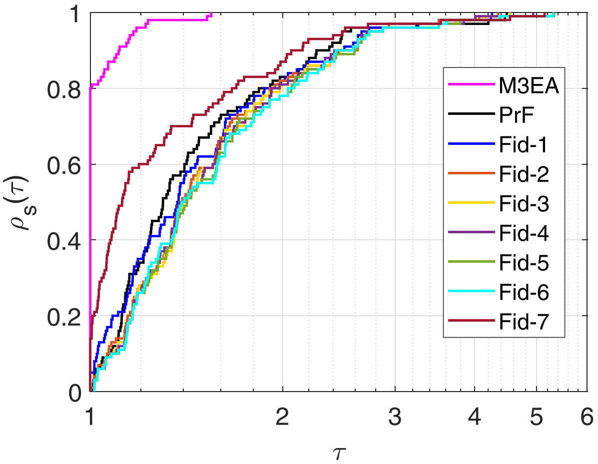

Figure 5. Performance profile plots of the competing algorithms considering the median IGD values of 25 independent runs.

is generated, its class can be predicted. If the SVM model predicts the label of an offspring solution to be 1 or likely to be an ND solution, it is considered as a promising solution. All promising ND solutions (O_{PND}) get a chance to compete against the parent or other promising solutions in the environmental selection process.

Anytime the archive is updated, we identify the ND solutions in the existing archive and train the SVM model with the ND (labelled 1) and the other solutions (labelled 0). Later, when a set of offspring solutions is generated in each iteration, we first predict the classes of the offspring solutions and only carry forward the ones which are labeled '1'. The rest of the algorithm remains unchanged. We refer to this enhanced algorithm M3EA-P (i.e., M3EA with pre-selection). The algorithm pseudo-code is presented in Algorithm 2 and the changes to the original M3EA are indicated with gray shaded boxes.

Algorithm 2 M3EA-P.

```
Input: ...
...
⋮
SVM = TrainSVM(A, A_ND)          ▷ Train only if pre-selection is enabled, ignore otherwise
...
while (cost < cost_max) do
    O_PND = GenerateOffsprings(μ, P)
    O = IdentifyPromising(O_PND, SVM) ▷ If pre-selection is enabled, identify promising ND solutions with the help of SVM
    ...
⋮
    SVM = TrainSVM(A, A_ND)
    ...
end while
Return A_ND.
```

The performance of the M3EA-P is compared with M3EA in Table 5 where the average IGD and the WRS test results are presented. In this table, we also list the total numbers of problem instances (n), wins (w), losses (l) and ties (t) of the M3EA when compared with M3EA-P. Further, the detailed statistics including the minimum, average and maximum IGD along with the WRS test restults are presented in Table VII of the supplementary file. As the classifier is acting as a fail-safe for the surrogate models and allowing only the potentially promising solutions to take part in the evolutionary process, the new variant is able to save significant evaluation cost. The resulting improvements are evident as the new variant shows statistically significantly better performance in 39 problem instances, while the original version is so performs better only on 5 problem instances among a total of 100 problem instances. The algorithms deliver equivalent results in 54 instances. Thus, among the two, M3EA-P is overall more likely to perform well for problems where the evaluation budget is very low. However, take note that the classifier model is introduced here to complement the surrogate assisted search and it can allow solutions that are not ND in the initial phase of the search. This can happen when the SVM model is trained with insufficient number of points which may result some unpromising solutions being selected. However, as the search progresses and the archive is updated, the SVM model is trained with more points and the prediction accuracy increases. Besides, even if an

Table 5
Test results for mean IGD metric obtained by M3EA-P and the original M3EA. The best mean results are highlighted in bold. Here, ↓, ↑ and ≈ indicates whether the particular algorithm is statistically significantly better, worse or equivalent to M3EA-P.

Prob.	Error Type	M	M3EA-P		M3EA
MFMaOP1	Resolution	2	72.7354	↓	**67.5418**
		3	**56.7942**	≈	**56.7942**
		6	**23.0774**	≈	**23.0774**
		8	**9.4851**	≈	**9.4851**
		10	**0.5404**	≈	**0.5404**
	Stochastic	2	**66.6029**	↑	76.3975
		3	58.9604	≈	**58.4183**
		6	**22.2276**	≈	**22.2276**
		8	**7.8852**	≈	**7.8852**
		10	**0.4965**	≈	**0.4965**
	Instability-1	2	**74.4468**	≈	76.7189
		3	**63.7239**	≈	**63.7239**
		6	27.8898	≈	**27.5776**
		8	**11.3031**	≈	**11.3031**
		10	**0.5398**	≈	**0.5398**
	Instability-2	2	**83.8415**	≈	89.6129
		3	**77.2374**	≈	**77.2374**
		6	**26.4237**	≈	**26.4237**
		8	**9.3225**	≈	**9.3225**
		10	**0.5071**	≈	**0.5071**
MFMaOP2	Resolution	2	0.2957	≈	**0.2365**
		3	**0.3346**	↑	0.5023
		6	**0.5396**	↑	0.6100
		8	**0.6304**	↑	0.6579
		10	0.6690	≈	**0.6681**
	Stochastic	2	0.2588	↓	**0.1944**
		3	**0.3263**	↑	0.4791
		6	**0.5076**	↑	0.5627
		8	**0.5888**	↑	0.6213
		10	**0.6346**	↑	0.6474
	Instability-1	2	**0.2887**	↑	0.3932
		3	0.4153	≈	**0.4149**
		6	**0.6311**	↑	0.7108
		8	**0.6368**	↑	0.6707
		10	**0.6452**	≈	**0.6452**
	Instability-2	2	0.2006	≈	**0.1906**
		3	**0.3247**	↑	0.4891
		6	**0.5180**	↑	0.5655
		8	**0.5853**	↑	0.6096
		10	**0.6372**	↑	0.6603
MFMaOP2^{-}	Resolution	2	**0.1760**	≈	0.1760
		3	0.4401	↓	**0.4223**
		6	**1.1146**	↑	1.1333
		8	**1.6450**	↑	1.6919
		10	**2.3431**	↑	2.3749
	Stochastic	2	4.1099	↓	**4.0328**
		3	**3.9566**	↑	4.0525
		6	**3.8010**	↑	3.8476
		8	**3.7179**	↑	3.7381
		10	**3.6765**	≈	3.6781
	Instability-1	2	**4.1098**	≈	4.1396
		3	**3.9763**	↑	4.1039
		6	**3.8446**	↑	3.8806
		8	**3.7299**	↑	3.7492
		10	**3.6750**	≈	3.6797
	Instability-2	2	4.0579	≈	**4.0359**
		3	**3.9416**	↑	4.0893
		6	**3.8210**	↑	3.8458
		8	**3.7167**	↑	3.7397
		10	**3.6762**	≈	3.6793

(*continued on next page*)

Table 5 (*continued*)

Prob.	Error Type	M	M3EA-P		M3EA
MFMaOP3	Resolution	2	**0.2614**	≈	0.2686
		3	**0.2341**	↑	0.2630
		6	**0.1293**	↑	0.1366
		8	0.0658	≈	**0.0650**
		10	**0.0140**	↑	0.0150
	Stochastic	2	0.1483	↓	**0.1219**
		3	**0.1707**	↑	0.1880
		6	**0.1070**	≈	0.1112
		8	**0.0600**	≈	0.0605
		10	**0.0141**	↑	0.0152
	Instability-1	2	0.1672	≈	**0.1618**
		3	**0.1811**	↑	0.2117
		6	**0.1139**	≈	0.1197
		8	0.0606	≈	**0.0601**
		10	**0.0142**	↑	0.0156
	Instability-2	2	**0.1191**	↑	0.2081
		3	**0.1642**	≈	**0.1642**
		6	**0.1037**	≈	**0.1037**
		8	**0.0553**	≈	**0.0553**
		10	**0.0139**	≈	0.0139
MFMaOP4	Resolution	2	**1.5849**	↑	2.1949
		3	**2.5962**	≈	**2.5962**
		6	**3.3932**	≈	**3.3932**
		8	**3.5756**	≈	**3.5756**
		10	**1.2148**	≈	**1.2148**
	Stochastic	2	**1.4592**	↑	2.2457
		3	**2.4887**	≈	**2.4887**
		6	**3.5045**	≈	**3.5045**
		8	**4.1187**	≈	**4.1187**
		10	**1.1883**	≈	**1.1883**
	Instability-1	2	**2.0545**	↑	2.5968
		3	**3.0317**	≈	**3.0317**
		6	**5.1094**	≈	**5.1094**
		8	**4.1355**	≈	**4.1355**
		10	**1.1810**	≈	**1.1810**
	Instability-2	2	**1.3170**	↑	2.4218
		3	**2.7861**	≈	**2.7861**
		6	**4.2636**	≈	**4.2636**
		8	**3.3170**	≈	**3.3170**
		10	**1.1830**	≈	**1.1830**
	n/w/l/t		–/–/–		100/39/5/56

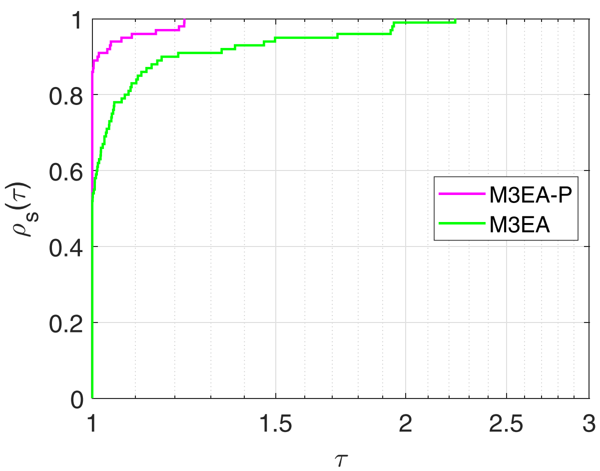

Figure 6. Performance profile plots of the two M3EA varinats with and without SVM classifier considering the median IGD values of 25 independent runs.

unpromising solution is selected at an early phase and the surrogate models can indicate its poor performance, the solution is filtered-out during the environmental selection process.

Finally, the performances of both the variants can be visually observed from the performance-profile plot in Figure 6. It shows the superiority of the enhanced *M3EA-P* over the original M3EA based on the median IGD values for 25 independent runs. This again reinforces the advantages of pre-selecting the promising solutions.

6. Conclusions and future research directions

In this paper, we propose an algorithm M3EA to solve computationally expensive multi/many-objective optimization problems with multiple fidelity levels, specifically geared towards iterative solvers. The algorithm extends the current capability in multi-fidelity optimization by enabling efficient solution to problems which contain multiple objectives and multiple fidelity levels. The paper also introduced a set of scalable, multi-objective, multi-fidelity benchmark problems with different types of Pareto fronts.

The performance of M3EA is tested on the proposed benchmark test problems. From the results, it is observed that the proposed algorithm is able to perform significantly better than two alternate strategies: equally dividing the evaluation budget to different fidelity levels and progressively running the optimization, individually performing with the help of different fidelity levels (including the highest fidelity level itself). The results indicate the algorithm's competence and reliability in dealing with a wide range of problems with various Pareto front shapes. This is promising and highlights its potential for real-life applications. Further enhancements in the algorithm are also demonstrated by incorporating pre-selection.

Some future research directions include extending the study to solve problems with constraints and studying the adaptation of reference vectors to deal with multi-fidelity optimization problems with irregular (e.g. degenerate, disconnected and highly non-linear) Pareto fronts.

Declaration of conflicts of interest

None.

Supplementary material

Supplementary material associated with this article can be found, in the online version, at 10.1016/j.ins.2019.06.016.

References

[1] H. Wang, Y. Jin, J. Doherty, A generic test suite for evolutionary multifidelity optimization, IEEE Trans. Evol. Comput. 22 (6) (2018) 836–850, doi:10.1109/TEVC.2017.2758360.

[2] A. Habib, H.K. Singh, T. Ray, A multiple surrogate assisted evolutionary algorithm for optimization involving iterative solvers, Eng. Optim. 50 (9) (2018) 1625–1644.

[3] J. Branke, M. Asafuddoula, K.S. Bhattacharjee, T. Ray, Efficient use of partially converged simulations in evolutionary optimization, IEEE Trans. Evol. Comput. 21 (1) (2017) 52–64.

[4] S. Choi, J.J. Alonso, I.M. Kroo, M. Wintzer, Multifidelity design optimization of low-boom supersonic jets, J. Aircr. 45 (1) (2008) 106–118.

[5] Y. Jin, Surrogate-assisted evolutionary computation: Recent advances and future challenges, Swarm Evol. Comput. 1 (2) (2011) 61–70.

[6] R.T. Haftka, Combining global and local approximations, AIAA J. 29 (9) (1991) 1523–1525.

[7] I.C. Kampolis, A.S. Zymaris, V.G. Asouti, K.C. Giannakoglou, Multilevel optimization strategies based on metamodel-assisted evolutionary algorithms, for computationally expensive problems, in: Proceedings of IEEE Congress on Evolutionary Computation (CEC), Suntec City, Singapore, 2007, pp. 4116–4123.

[8] M.K. Zahir, Z. Gao, Variable-fidelity optimization with design space reduction, Chin. J. Aeronaut. 26 (4) (2013) 841–849.

[9] J.W. Bandler, Q.S. Cheng, S.A. Dakroury, A.S. Mohamed, M.H. Bakr, K. Madsen, J. Sondergaard, Space mapping: the state of the art, Microwave Theory and Techniques, IEEE Trans. Microw. Theory Tech. 52 (1) (2004) 337–361.

[10] Z. Qian, C.C. Seepersad, V.R. Joseph, J.K. Allen, C.F.J. Wu, Building surrogate models based on detailed and approximate simulations, J. Mech. Des. 128 (4) (2006) 668–677.

[11] G. Sun, G. Li, S. Zhou, W. Xu, X. Yang, Q. Li, Multi-fidelity optimization for sheet metal forming process, Struct. Multidiscip. Optim. 44 (1) (2011) 111–124.

[12] A. March, K. Willcox, Constrained multifidelity optimization using model calibration, Struct. Multidiscip. Optim. 46 (1) (2012) 93–109.

[13] G. Sun, G. Li, Q. Li, Variable fidelity design based surrogate and artificial bee colony algorithm for sheet metal forming process, Finite. Elem. Anal. Des. 59 (0) (2012) 76– 90.

[14] M.C. Kennedy, A. O'Hagan, Predicting the output from a complex computer code when fast approximations are available, Biometrika 87 (1) (2000) 1–13.

[15] S.-H. Son, D.-H. Park, K.-J. Chan, D.-H. Choi, Constrained global gesign optimization using a multi-fidelity model, in: Proceedings of 10th World Congress on Structural and Multidisciplinary Optimization, Orlando, Florida, USA, 2013.

[16] B. Liu, S. Koziel, Q. Zhang, A multi-fidelity surrogate model-assisted evolutionary algorithm for computationally expensive optimization problems, J. Comput. Sci. 12 (2) (2016) 28–37.

[17] P.H. Reisenthel, T.T. Allen, D.J. Lesieutre, S.H. Lee, Development of Multidisciplinary, Multifidelity Analysis, Integration and Optimization of Aerospace Vehicles, Technical Report, Defense Technical Information Center, US Department of Defense, 2010.

[18] D. Lim, Y.-S. Ong, Y. Jin, B. Sendhoff, Evolutionary optimization with dynamic fidelity computational models, in: Advanced Intelligent Computing Theories and Applications. With Aspects of Artificial Intelligence, in: Lecture Notes in Computer Science, vol. 5227, Springer Berlin Heidelberg, 2008, pp. 235–242.

[19] L. Le Gratiet, C. Cannamela, Cokriging-based sequential design strategies using fast cross-validation techniques for multi-fidelity computer codes, Technometrics 57 (3) (2015) 418–427.

[20] K.S. Bhattacharjee, H.K. Singh, M. Ryan, T. Ray, Bridging the gap: Many-objective optimization and informed decision-making, IEEE Trans. Evol. Comput. 21 (5) (2017) 813–820.

[21] Q. Zhang, W. Liu, E. Tsang, B. Virginas, Expensive multiobjective optimization by moea/d with gaussian process model, IEEE Trans. Evol. Comput. 14 (3) (2010) 456–474.
[22] J. Knowles, Parego: A hybrid algorithm with on-line landscape approximation for expensive multiobjective optimization problems, IEEE Trans. Evol. Comput. 10 (1) (2006) 50–66.
[23] L. Pan, C. He, C. He, Y. Tian, H. Wang, X. Zhang, Y. Jin, A classification based surrogate-assisted evolutionary algorithm for expensive many-objective optimization, IEEE Trans. Evol. Comput. doi:10.1109/TEVC.2018.2802784.
[24] T. Chugh, Y. Jin, K. Miettinen, J. Hakanen, K. Sindhya, A surrogate-assisted reference vector guided evolutionary algorithm for computationally expensive many-objective optimization, IEEE Trans. Evol. Comput. 22 (1) (2018) 129–142.
[25] T. Chugh, K. Sindhya, J. Hakanen, K. Miettinen, A survey on handling computationally expensive multiobjective optimization problems with evolutionary algorithms, Soft Computing. doi:10.1007/s00500-017-2965-0.
[26] M. Drela, Pros & cons of airfoil optimization, in: Frontiers of Computational Fluid Dynamics 1998, World Scientific, 1998, pp. 363–381.
[27] M. Nemec, D.W. Zingg, T.H. Pulliam, Multipoint and multi-objective aerodynamic shape optimization, AIAA J. 42 (6) (2004) 1057–1065.
[28] K. Deb, L. Thiele, M. Laumanns, E. Zitzler, Scalable Test Problems for Evolutionary Multi-Objective Optimization, Springer, 2005.
[29] Ö. Öksüz, İ. S. Akmandor, Multi-objective aerodynamic optimization of axial turbine blades using a novel multilevel genetic algorithm, J. Turbomach. 132 (4) (2010) 041009.
[30] S. Koziel, S. Ogurtsov, Multi-objective design of antennas using variable-fidelity simulations and surrogate models, IEEE Trans. Antennas Propag. 61 (12) (2013) 5931–5939.
[31] J. Sacks, W.J. Welch, T.J. Mitchell, H.P. Wynn, et al., Design and analysis of computer experiments, Stat. Sci. 4 (4) (1989) 409–423.
[32] J. Zhu, Y.-J. Wang, M. Collette, A multi-objective variable-fidelity optimization method for genetic algorithms, Eng. Optim. 46 (4) (2014) 521–542.
[33] Q. Zhou, Y. Wang, S.-K. Choi, P. Jiang, An on-line multi-fidelity metamodel assisted multi-objective genetic algorithm, in: Proceedings of the ASME International Design Engineering Technical Conferences and Computers and Information in Engineering Conference, 2017, pp. 1–10.
[34] K. Deb, S. Agrawal, A. Pratap, T. Meyarivan, A fast elitist non-dominated sorting genetic algorithm for multi-objective optimization: Nsga-ii, Lect. Notes Comput. Sci. 1917 (2000) 849–858.
[35] R. Cheng, Y. Jin, M. Olhofer, B. Sendhoff, A reference vector guided evolutionary algorithm for many-objective optimization, IEEE Trans. Evol. Comput. 20 (5) (2016) 773–791.
[36] I. Das, J.E. Dennis, Normal-bounday intersection: A new method for generating Pareto optimal points in multicriteria optimization problems, SIAM Journal on Optimization 8 (3) (1998) 631–657.
[37] K. Deb, H. Jain, An evolutionary many-objective optimization algorithm using reference-point based non-dominated sorting approach, Part I: solving problems with box constraints, IEEE Trans. Evol. Comput. 18 (4) (2014) 577–601.
[38] R. Storn, K. Price, Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces, J. Global Optim. 11 (4) (1997) 341–359.
[39] R.H. Myers, D.C. Montgomery, et al., Response Surface Methodology: Process and Product Optimization Using Designed Experiments, vol. 3, Wiley New York, 1995.
[40] J. Park, I.W. Sandberg, Universal approximation using radial-basis-function networks, Neural Comput. 3 (2) (1991) 246–257.
[41] H. Ishibuchi, Y. Setoguchi, H. Masuda, Y. Nojima, Performance of decomposition-based many-objective algorithms strongly depends on Pareto front shapes, IEEE Trans. Evol. Comput. 21 (2) (2017) 169–190.
[42] M.A. El-Beltagy, A. Keane, A comparison of various optimization algorithms on a multilevel problem, Eng. Appl. Artif. Intell. 12 (5) (1999) 639–654.
[43] P.A.N. Bosman, D. Thierens, The balance between proximity and diversity in multiobjective evolutionary algorithms, IEEE Trans. Evol. Comput. 7 (2) (2003) 174–188.
[44] Y. Tian, R. Cheng, X. Zhang, F. Cheng, Y. Jin, An indicator based multi-objective evolutionary algorithm with reference point adaptation for better versatility, IEEE Trans. Evol. Comput. doi:10.1109/TEVC.2017.2749619.
[45] F. Wilcoxon, S.K. Katti, R.A. Wilcox, Critical Values and Probability Levels for the Wilcoxon Rank Sum Test and the Wilcoxon Signed Rank Test, American Cyanamid Company, 1963.
[46] E.D. Dolan, J.J. Moré, Benchmarking optimization software with performance profiles, Math. Program. 91 (2) (2002) 201–213.
[47] N. Cristianini, J. Shawe-Taylor, et al., An Introduction to Support Vector Machines and Other Kernel-Based Learning Methods, Cambridge University Press, 2000.