

An Evolutionary Approach to the Solution of Multi-Objective Min-Max Problems in Evidence-Based Robust Optimization

Simone Alicino and Massimiliano Vasile

Abstract—This paper presents an evolutionary approach to solve the multi-objective min-max problem (MOMMP) that derives from the maximization of the Belief in robust design optimization. In evidence-based robust optimization, the solutions that minimize the design budgets are robust under epistemic uncertainty if they maximize the Belief in the realization of the value of the design budgets. Thus robust solutions are found by minimizing, with respect to the design variables, the global maximum with respect to the uncertain variables. This paper presents an algorithm to solve MOMMP, and a computational cost reduction technique based on Kriging metamodels. The results show that the algorithm is able to accurately approximate the Pareto front for a MOMMP at a fraction of the computational cost of an exact calculation.

I. INTRODUCTION

GIVEN the model of a system or process, the lower expectation in the realization of the value of a particular performance index for that system or process can be defined as the degree of belief that one has in a certain proposition being true, given the available evidence. In the framework of imprecise probabilities, it can be seen as a lower bound to the cumulative distribution function of classical probability theory. Its use is therefore interesting in engineering design, as it gives the lower limit of the confidence that the design budgets under uncertainty will be below a given threshold. In this framework both epistemic and aleatory uncertainties can be treated even when no exact information on the probability distribution associated to a particular uncertain quantity is available. Stochastic variables, with associated probability, are replaced by a multivalued mapping from a collection of subsets of an uncertain space U into an envelope defined by the lower and upper expectation (Belief and Plausibility in the case of Evidence Theory) in the realization of a particular value.

In the preliminary design of an engineering system the type of uncertainty is often mainly epistemic in nature, as in a later stage of the design more information is generally available. It is therefore natural to assign belief masses to intervals of values rather than precise probabilities. The main drawback with the use of multivalued mappings is that the computation of the lower and upper expectations, Belief and Plausibility in the case of belief functions, has a complexity that is exponential with the number of uncertain variables. Recently, Vasile et al. [1] proposed some

strategies to obtain an estimation of the maximum Belief with a reduction of the computational cost. The approach starts by translating an optimization under uncertainty into a single or multi-objective min-max problem equivalent to a worst-case scenario optimization problem. Although this approach avoids the intrinsic exponential complexity in the computation of the Belief, it still requires a high number of function evaluations. This paper presents an algorithm that implements special heuristics to increase the probability to find the exact global Pareto front in the case of a min-max problem on multi-modal functions. Also, it presents an approach based on Kriging surrogate models to reduce the computational cost associated to the solution of the MOMMP. The combination of evolutionary algorithms and surrogate models has been object of considerable research efforts in the last decade, as summarized in [2], [3]. However, to the authors' knowledge, not many papers deal with surrogate-based min-max optimization. Ideas of combining evolutionary optimization and surrogate modelling to solving min-max problems were proposed in [4], [5]. More recently Marzat et al. [6] proposed a non-evolutionary approach using Kriging metamodels to reduce the computational cost of the solution of single objective min-max problem in worst-case scenario optimization.

The approach proposed in this paper addresses specifically multi-objective min-max optimization problems and differs from previous work in the way the surrogate model interfaces with the optimization algorithm. As it will be shown, the ultimate goal is to build a surrogate of the maximization process. The paper starts with a brief introduction to Evidence Theory and its use in the context of robust design optimization in Section II. Section III introduces an algorithm to compute a multi-objective optimal design solution under uncertainty. Section IV explains the use of surrogate modelling to reduce the computational cost in multi-objective optimization cases. Section V presents the results on some scalable test cases that were applied to assess the performance of the proposed algorithm.

II. EVIDENCE-BASED ROBUST DESIGN OPTIMIZATION

Evidence Theory (also known as Dempster-Shafer Theory) [7] allows to adequately model both epistemic and aleatory uncertainty when no information on the probability distributions is available. For instance, during the preliminary design of an engineering system, experts can provide informed opinions by expressing their belief in an uncertain parameter u being within a certain set of intervals. The level of confidence an expert has in u belonging to one of the intervals is

Simone Alicino and Massimiliano Vasile are with the Department of Mechanical and Aerospace Engineering, University of Strathclyde, Glasgow, United Kingdom (email: {simone.alicino, massimiliano.vasile}@strath.ac.uk).

This work is partially supported by an ESA/NPI grant 'Evidence-based Robust Design Optimization'

quantified by using a mass function generally known as Basic Probability Assignment (*bpa*). Note that the *bpa* is a belief rather than an actual probability. All the intervals form the so-called frame of discernment Θ , which is a set of mutually exclusive elementary propositions. The frame of discernment can be viewed as the counterpart of the finite sample space in probability theory. The power set of Θ is called $U = 2^\Theta$, or the set of all the subsets of Θ (the uncertain space in the following). An element θ of U that has a non-zero *bpa* is called focal element. When more than one parameter is uncertain, the focal elements are the result of the Cartesian product of all the elements of each power set associated to each uncertain parameter. The *bpa* of a given focal element is then the product of the *bpa* of all the elements in the power set associated to each parameter. All the pieces of evidence completely in support of a given proposition form the cumulative belief function *Bel*, whereas all the pieces of evidence partially in support of a given proposition form the cumulative plausibility function *Pl*. The Belief *Bel* and the Plausibility *Pl* functions are defined as follows:

$$Bel(A) = \sum_{\forall \theta_i \subseteq A} m(\theta_i) \quad (1)$$

$$Pl(A) = \sum_{\forall \theta_i \cap A \neq \emptyset} m(\theta_i) \quad (2)$$

where A is the proposition about which Belief and Plausibility need to be evaluated. For example, the proposition can be expressed as:

$$A = \{\mathbf{u} \in U \mid f(\mathbf{u}) \leq \nu\} \quad (3)$$

where f is the outcome of the system model and the threshold ν is the desired value of a design budget (e.g. the mass). Thus, focal elements intercepting the set A , but not fully included in it, are considered in *Pl* but not in *Bel*. It is important to note that the set A can be disconnected or present holes, likewise the focal elements can be disconnected or partially overlapping.

An engineering system to be optimized can be modelled as a function $f : D \times U \subseteq \mathbb{R}^{m+n} \rightarrow \mathbb{R}$. The function f represents the model of the system budgets (e.g. power budget, mass budget, etc.), and depends on some uncertain parameters $\mathbf{u} \in U$ and design parameters $\mathbf{d} \in D$, where D is the available design space and U the uncertain space. What is interesting for the designers is the value of the function f for which $Bel = 1$, i.e. it is maximum. This value of the design budget is the threshold ν_{\max} above which the design is certainly feasible, given the current body of evidence. If q objective functions exist, then the following problem can be solved without considering all the focal elements:

$$\nu_{\max} = \min_{\mathbf{d} \in D} \mathbf{F} = \min_{\mathbf{d} \in D} [\max_{\mathbf{u} \in U} f_1(\mathbf{d}, \mathbf{u}), \dots, \max_{\mathbf{u} \in U} f_q(\mathbf{d}, \mathbf{u})]^T \quad (4)$$

Problem in 4 is a multi-objective min-max over the design space D and the uncertain space \bar{U} , where \bar{U} is a unit hypercube collecting all the focal elements in a compact set

with no overlapping or holes. The transformation between U and \bar{U} is given by:

$$\mathbf{x}_U = \frac{(\mathbf{b}_{U,i}^u - \mathbf{b}_{U,i}^l)}{(\mathbf{b}_{\bar{U},i}^u - \mathbf{b}_{\bar{U},i}^l)} \mathbf{x}_{\bar{U},i} + \mathbf{b}_{U,i}^l - \frac{(\mathbf{b}_{U,i}^u - \mathbf{b}_{U,i}^l)}{(\mathbf{b}_{\bar{U},i}^u - \mathbf{b}_{\bar{U},i}^l)} \mathbf{b}_{\bar{U},i}^l \quad (5)$$

where $\mathbf{b}_{U,i}^u$ and $\mathbf{b}_{U,i}^l$ (resp. $\mathbf{b}_{\bar{U},i}^u$ and $\mathbf{b}_{\bar{U},i}^l$) are the upper and lower boundaries of the i -th hypercube to which $\mathbf{x}_{U,i}$ (resp. $\mathbf{x}_{\bar{U},i}$) belongs.

III. A MULTI-OBJECTIVE MIN-MAX EVOLUTIONARY ALGORITHM

Problem in 4 searches for the minimum of the maxima of all the functions over \bar{U} and represents an example of worst-case scenario design optimization. The maximum of every function is independent of the other functions and corresponds to a different uncertain vector. Therefore, all the maxima can be computed in parallel with q single-objective maximizations. The maximization of each function is performed by running a global optimization over \bar{U} using Inflationary Differential Evolution (IDEA). The minimization over D is performed by means of MACS ν , a modified version of MACS2.

IDEA [8] is a population-based memetic algorithm for single-objective optimization. It hybridizes Differential Evolution and Monotonic Basin Hopping paradigms in order to simultaneously improve local convergence and avoid stagnation, as demonstrated for some space trajectory optimization problems.

MACS2 [9] is a memetic algorithm for multi-objective optimization based on a combination of Pareto ranking and Tchebycheff scalarization. The search for non-dominated solutions is performed by a population of agents which combine individualistic and social actions. The initial population is randomly generated in the search domain D . Individualistic actions perform a sampling of the search space in a neighborhood N_ρ of each agent. Then, subsets of the population perform social actions aiming at following particular descent directions in the criteria space. Social agents implement a Differential Evolution operator and assess the new candidate solutions using Tchebycheff scalarization. Current non-dominated solutions are then stored in an archive. Both social and individualistic actions make use of a combination of the population and the archive.

MACS ν is the min-max variant of MACS2. Indeed, in a classical minimization problem two solutions \mathbf{d}_1 and \mathbf{d}_2 are ranked according to which one gives the lower value of the function. In the minimization loop of a min-max problem, the same can be done only if the maximization loop has returned the actual global maxima $\bar{\mathbf{u}}_1$ and $\bar{\mathbf{u}}_2$. However, this is usually not true. Therefore a mechanism of cross-check such that also $(\mathbf{d}_1, \mathbf{u}_2)$ and $(\mathbf{d}_2, \mathbf{u}_1)$ are evaluated is needed in order to increase the probability that each maximization identifies the global maximum and correctly rank two solutions. For this reason, MACS ν (see Algorithm 1) endows MACS2 with special heuristics. More in detail, Algorithm 2 is used to discriminate and archive the solutions that are Pareto

dominant. A cross check is necessary to compare the values of the objective functions for a newly generated design vector against the function values of an already archived solution. Consider, in fact, that a different design vector corresponds in general to a different landscape of the objective functions, and therefore to a different location of the maxima. Moreover, the cross-check performs a local search or a simple function evaluation in the inner maximization loop depending on whether the location of the maxima changes or not, respectively, for different design vectors.

At the end of this cross-check, Algorithm 3 is run to select the design vectors to attribute to the next generation, once a new candidate population is generated after either individualistic or social moves. The following heuristics are implemented: if \mathbf{d} (resp. \mathbf{u}) is unchanged, the old \mathbf{u} (resp. \mathbf{d}) is replaced with the new one, if it yields a better value of the objective function; if both \mathbf{d} and \mathbf{u} are different, the new vectors will replace the old ones.

A validation (see Algorithm 4) is run at the last iteration of MACS ν , after the individualistic and social moves have been performed. It performs a global search for the extremes in the archive, and replaces the corresponding uncertain vector and objective if the new ones give a better value of the objective function, until there is no more variation in the extremes of the archive. This step is introduced to mitigate the possibility that the cross-check operators assign the same incorrect \mathbf{u} to all \mathbf{d} vectors in the population and archive.

A. Optimization Performance Metrics

In order to assess the Pareto front f found by MACS ν with respect to a reference front g , we made use of convergence M_{conv} (Equation 6), and spreading M_{spr} (Equation 7), defined as:

$$M_{conv} = \frac{1}{N_p} \sum_{i=1}^{N_p} \min_{j \in M_p} 100 \left\| \frac{g_j - f_i}{g_j} \right\| \quad (6)$$

$$M_{spr} = \frac{1}{M_p} \sum_{i=1}^{M_p} \min_{j \in N_p} 100 \left\| \frac{f_j - g_i}{g_i} \right\| \quad (7)$$

where N_p and M_p are the cardinality of the fronts f and g , respectively. As one can see, the lower the value of M_{conv} and M_{spr} , the better convergence and spreading, respectively. In addition, two performance indexes $p_{conv} = P(M_{conv} < t_{conv})$ and $p_{spr} = P(M_{spr} < t_{spr})$ compute the success rate with respect to some thresholds t_{conv} and t_{spr} . These indexes tell how many times, over a certain number of runs, convergence and spreading are equal to or below such thresholds.

IV. SURROGATE MODELLING

The solution of the min-max problem involves an increase in the number of times the objective function needs to be evaluated. In this context, the use of surrogate models can play a valuable role. The surrogates are constructed using data drawn from high-fidelity models, and provide fast approximations of the objectives at new design points. In this

Algorithm 1 MACS ν

```

1: Initialize population  $P$ , archive  $A = P$ ,  $n_{feval} = 0$ ,
    $\epsilon = 0.7$ ,  $\delta = 10^{-6}$ 
2: while  $n_{feval} < n_{feval,max}$ 
3:   Run individualistic moves and generate trial population  $P_t$ 
4:   for all  $\mathbf{d} \in P_t$ 
5:     for all  $\mathbf{d}_{arch} \in A$ 
6:       if  $\mathbf{d} \succ \mathbf{d}_{arch}$ 
7:         CROSS-CHECK( $P_t, A$ ) ▷ Alg. 2
8:       end if
9:     end for
10:  end for
11:  MIN-MAX SELECTION( $P, P_t$ ) ▷ Alg. 3
12:  Update  $P$  and  $A$ 
13:   $Z \leftarrow \|\mathbf{F}_{arch}^{max} - \mathbf{F}_{arch}^{min}\|$ 
14:  Run social moves and generate candidate population  $P_s$ 
15:  for all  $\mathbf{d} \in P_s$ 
16:    for all  $\mathbf{d}_{arch} \in A$ 
17:      if  $\mathbf{d} \succ \mathbf{d}_{arch}$  or  $\|\mathbf{F}(\mathbf{d}) - \mathbf{F}(\mathbf{d}_{arch})\| > \epsilon Z$ 
18:        CROSS-CHECK( $P_s, A$ ) ▷ Alg. 2
19:      end if
20:    end for
21:  end for
22:  MIN-MAX SELECTION( $P, P_s$ ) ▷ Alg. 3
23:  Update  $P$  and  $A$ 
24:  VALIDATION( $A$ ) ▷ Alg. 4
25:  for all  $\mathbf{d} \in P$ 
26:    for all  $\mathbf{d}_{arch} \in A$ 
27:      if  $\mathbf{d} \succ \mathbf{d}_{arch}$  or  $\mathbf{d} \prec \mathbf{d}_{arch}$ 
28:        CROSS-CHECK( $P, A$ ) ▷ Alg. 2
29:      else if  $\|\mathbf{F}(\mathbf{d}) - \mathbf{F}(\mathbf{d}_{arch})\| < \delta$ 
30:        Replace  $\mathbf{u} \in P$  with  $\mathbf{u} \in A$ 
31:      end if
32:    end for
33:  end for
34:  Fit surrogate model in  $D$  with the elements of  $A$ 
35: end while

```

paper we propose an approach in which a surrogate is built in the D space only, as shown in Figure 1(a). The surrogate in the design space has \mathbf{d} as design sites, and g as response, and is therefore constructed and updated in the outer loop of the optimization, i.e. the minimization over D .

The approach can be formalized as follows. Let us consider, without loss of generality, Problem 4 in the single-objective case:

$$\nu_{\max} = \min_{\mathbf{d} \in D} \max_{\mathbf{u} \in U} f(\mathbf{d}, \mathbf{u}) \quad (8)$$

This can also be written as

$$\nu_{\max} = \min_{\mathbf{d} \in D} g(\mathbf{d}, \mathbf{u}) \quad (9)$$

where

$$g(\mathbf{d}, \mathbf{u}) = \max_{\mathbf{u} \in U} f(\mathbf{d}, \mathbf{u}) \quad (10)$$

Algorithm 2 Cross-check

```

1: function CROSS-CHECK( $P, A$ )
2:   for all  $l \in \{1, \dots, q\}$ 
3:     if  $\|\mathbf{u}_{arch}^l - \mathbf{u}^l\| \neq 0$ 
4:       Compute local maxima  $\tilde{\mathbf{u}}_{arch}^l$  and  $\tilde{\mathbf{u}}^l$  associated to  $\mathbf{d}$  and  $\mathbf{d}_{arch}$ 
5:       if  $f^l(\mathbf{d}_{arch}, \tilde{\mathbf{u}}^l) \geq f^l(\mathbf{d}_{arch}, \mathbf{u}_{arch}^l)$ 
6:         replace  $\mathbf{u}_{arch}^l$  with  $\tilde{\mathbf{u}}^l$ 
7:       end if
8:       if  $f^l(\mathbf{d}, \tilde{\mathbf{u}}_{arch}^l) \geq f^l(\mathbf{d}, \mathbf{u}^l)$ 
9:         replace  $\mathbf{u}^l$  with  $\tilde{\mathbf{u}}_{arch}^l$ 
10:      end if
11:    end if
12:  end for
13:  return  $P, A$ 
14: end function

```

Algorithm 3 Min-Max Selection

```

1: function MIN-MAX SELECTION( $P, P_{new}$ )
2:   for all  $l \in \{1, \dots, q\}$ 
3:     for all  $(\mathbf{d} \in P) \cup (\mathbf{d}_{new} \in P_{new})$ 
4:       if  $\|\mathbf{d}_{new} - \mathbf{d}\| = 0$ 
5:         if  $f_{new}^l(\mathbf{d}_{new}, \mathbf{u}_{new}^l) \geq f^l(\mathbf{d}, \mathbf{u}^l)$ 
6:           replace  $\mathbf{u}^l$  with  $\mathbf{u}_{new}^l$ 
7:           replace  $f_l$  with  $f_{new}^l$ 
8:         end if
9:       else if  $\|\mathbf{d}_{new} - \mathbf{d}\| > 0$ 
10:        if  $f_{new}^l(\mathbf{d}_{new}, \mathbf{u}_{new}^l) < f^l(\mathbf{d}, \mathbf{u}^l)$ 
11:          replace  $\mathbf{d}$  with  $\mathbf{d}_{new}$ 
12:          replace  $f_l$  with  $f_{new}^l$ 
13:        end if
14:      end if
15:    end for
16:  end for
17:  return  $P$ 
18: end function

```

i.e. it is the result of the inner loop of the optimization.

The aim of this approach is to separate the outer and inner loops. The idea behind this is that the inner loop can be called only if the surrogate (\mathbf{d}, g) needs to be updated. If the accuracy of the surrogate (\mathbf{d}, g) is above a certain threshold, only the outer loop is run, hence saving computational expense. In addition, such approach allows the surrogate model to be dependent only on the relevant parameters, i.e. the design vector \mathbf{d} for the outer loop (Figure 1(b)).

The surrogate model is built by using the archived solutions of the minimization loop as design sites (see Algorithm 1). The surrogate is therefore built and updated only if there are new elements in the archive. Note that a strategy that would build the surrogate progressively as new agents are deployed into the minimization loop, therefore using all the design solutions is not applicable because the design solutions need to be cross-checked. Without these steps, the surrogate would contain design solutions that, while being

Algorithm 4 Validation

```

1: function VALIDATION( $A$ )
2:   for all  $l \in \{1, \dots, q\}$ 
3:      $\Delta f_{best} = 1$ 
4:     while  $\Delta f_{best} \neq 0$ 
5:        $j = \text{argmin } f^l \in A$ 
6:       Run global optimization over  $\bar{U}$  and compute new  $\bar{f}^l$  and associated  $\bar{\mathbf{u}}^l$ 
7:       if  $\bar{f}^l > f^l$ 
8:         replace  $\mathbf{u}^l \in A$  with  $\bar{\mathbf{u}}^l$ 
9:         replace  $f^l \in A$  with  $\bar{f}^l$ 
10:         $\Delta f_{best} = \bar{f}^l - f^l$ 
11:      end if
12:    end while
13:  end for
14:  return  $A$ 
15: end function

```

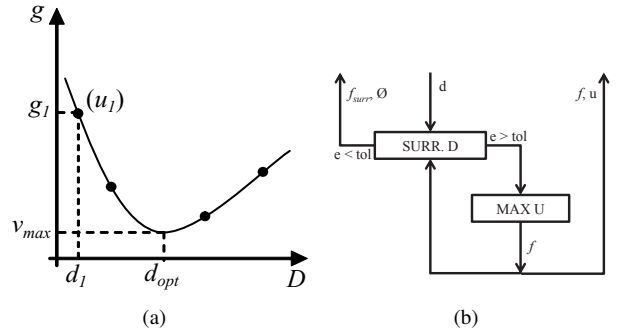


Fig. 1

(A) CONCEPTUAL EXAMPLE OF THE SURROGATE MODELLING STRATEGY. THE PLOT REPRESENTS THE SURROGATE IN THE D SPACE. THE DOTS ARE THE DESIGN SITES. NOTICE THAT $g_1 = \max_u f(d_1, u)$.

(B) SCHEMATIC OF THE SURROGATE MODELLING STRATEGY.

non-dominated, do not maximize the inner loop. A further feature of our technique is that the number of design sites is kept below a user-specified threshold of n_s points. Once the number of sample points overcomes such threshold, the sites to be retained are chosen so that the spreading of the Pareto front is maximized, and the members of the archive evenly distributed. By noting that the design sites are, as explained in the previous paragraph, the current archive of Pareto set and front, therefore one can define a sector in the criteria space defined by the origin and the extrema of the front. Then, such sector is divided into $n_s - 1$ smaller sectors of equal angle, and the design sites to be retained are the n_s ones which distance to the boundaries between the sectors is minimum. This allows for the surrogate to approximate accurately the current Pareto front, and also to keep the surrogate to a reasonable size.

A. Kriging Predictor

Starting from a set of sample points, or design sites, the Kriging predictor is an interpolation technique that makes

use of a regression function and a correlation model to predict the response of a function at a desired point. Being an interpolation method, it gives an exact prediction of the response at the sample points. Moreover, it assumes that the output function values are correlated in design space, i.e. closer points are more highly correlated. A complete derivation of the Kriging model can be found in Jones [10]. If we suppose to have a set of n design sites (\mathbf{x}, \mathbf{y}) , the correlation matrix \mathbf{R} of the design points can be expressed as

$$\mathbf{R} = [R_{ij}] = \exp \left[- \sum_{l=1}^d \theta_l \|\mathbf{x}_{il} - \mathbf{x}_{jl}\|^{p_l} \right] \quad (11)$$

where $i, j = 1, \dots, n$. In the same way, the correlation of the new point \mathbf{x}^* at which we want to predict the response $\hat{y}(\mathbf{x}^*)$, with the design points can be expressed as

$$\mathbf{r} = [r_i] = \exp \left[- \sum_{l=1}^d \theta_l \|\mathbf{x}_l^* - \mathbf{x}_{il}\|^{p_l} \right] \quad (12)$$

The correlation model depends on two parameters θ and p . They can be found to be the ones that minimize the mean squared error between the predicted response \hat{y} and the actual response y . Therefore an optimization is to be performed during the training phase. Under the assumption that the regression function is a zero-th order polynomial, i.e. it is a $n \times 1$ vector of ones $\mathbf{1}$, the prediction $\hat{y}(\mathbf{x}^*)$ can be found to be

$$\hat{y}(\mathbf{x}^*) = \hat{\mu} + \mathbf{r}^T \mathbf{R}^{-1} (\mathbf{y} - \mathbf{1} \hat{\mu}) \quad (13)$$

where

$$\hat{\mu} = \frac{\mathbf{1}^T \mathbf{R}^{-1} \mathbf{y}}{\mathbf{1}^T \mathbf{R}^{-1} \mathbf{1}} \quad (14)$$

One of the key benefits of Kriging is the provision of an estimated error of its predictions. The estimated mean squared error (MSE) for a Kriging model is

$$s^2(\mathbf{x}^*) = \hat{\sigma}^2 \left[1 - \mathbf{r}^T \mathbf{R}^{-1} \mathbf{r} + \frac{(1 - \mathbf{r}^T \mathbf{R}^{-1} \mathbf{r})^2}{\mathbf{1}^T \mathbf{R}^{-1} \mathbf{1}} \right] \quad (15)$$

where

$$\hat{\sigma}^2 = \frac{(\mathbf{y} - \mathbf{1} \hat{\mu})^T \mathbf{R}^{-1} (\mathbf{y} - \mathbf{1} \hat{\mu})}{n} \quad (16)$$

is the estimated variance.

The availability of an estimate of the prediction error is a very convenient feature, as it can be exploited in the surrogate update strategy. Letting y_{min} be the current best function value, and $Y(\mathbf{x})$ a random variable normally distributed, with mean $\hat{y}(\mathbf{x})$ and standard deviation $s(\mathbf{x})$, describing the uncertainty about the function's value, an improvement I is achieved if $I(\mathbf{x}) = y_{min} - \hat{y}(\mathbf{x}) > 0$. In this paper we consider a method based on the probability of improvement, that is deemed to be one of the best two-stage methods to the Kriging update [10] [11]. The main advantage of this method is that the probability of improvement is independent of magnitude and units of the function value.

The probability of an improvement is the area of the PDF, with mean $\hat{y}(\mathbf{x})$ and standard deviation $s(\mathbf{x})$, calculated from y_{min} to $-\infty$. That is

$$P(I(\mathbf{x})) = \Phi(u(\mathbf{x})) \quad (17)$$

where Φ is the normal cumulative distribution function, and

$$u(\mathbf{x}) = \frac{y_{min} - \hat{y}(\mathbf{x})}{s(\mathbf{x})} = \frac{I(\mathbf{x})}{s(\mathbf{x})} \quad (18)$$

One can set a threshold τ on this probability of improvement, and update the Kriging predictor if $P(I) \geq \tau$. The advantage of such method is that the threshold τ is a simple non-dimensional probability measure, therefore independent of the magnitude and units of the function value.

The probability of improvement can be extended to the case of multiple objectives. In this case, an improvement is achieved when the new point dominates at least one member of the archived non-dominated points. Considering two uncorrelated objectives, the improvement $I(\mathbf{x})$ is then defined as $y_{1P} - \hat{y}_1(\mathbf{x}) > 0 \cup y_{2P} - \hat{y}_2(\mathbf{x}) > 0$, where y_{1P} and y_{2P} represent the members of the archived Pareto front. If the Pareto front is composed of N_P members, by defining

$$u_1^i(\mathbf{x}) = \frac{y_{1P}^i - \hat{y}_1(\mathbf{x})}{s_1(\mathbf{x})}, \quad u_2^i(\mathbf{x}) = \frac{y_{2P}^i - \hat{y}_2(\mathbf{x})}{s_2(\mathbf{x})} \quad (19)$$

for $i = 1, \dots, N_P$, the probability of improvement is

$$P(I(\mathbf{x})) = \Phi(u_1^1(\mathbf{x})) + \sum_{i=1}^{N_P-1} [\Phi(u_1^{i+1}(\mathbf{x})) - \Phi(u_1^i(\mathbf{x}))] \\ \times \Phi(u_2^{i+1}(\mathbf{x})) + [1 - \Phi(u_1^{N_P}(\mathbf{x}))] \Phi(u_2^{N_P}(\mathbf{x})) \quad (20)$$

This expression can be obtained by integrating the Gaussian distribution underlying the improvement I for the two objective functions between $-\infty$ and the Pareto front [12].

V. TEST CASE

The surrogate-based optimization algorithm described in the previous sections was tested on six bi-objective test cases. The chosen functions, reported in Table I, have the property of being easily scalable. More in detail, functions MV9, MV10, and EM1 present very challenging landscapes, with multiple maxima that can change significantly with the design vector. Function MV10, in particular, is characterized by having the maxima located on top of multiple sharp, steep peaks. The six test cases are composed of the functions in Table I as reported in Table II. The dimension of the design vector \mathbf{d} , as well as the uncertain vector \mathbf{u} , is 8, for a total of 16 dimensions, for test cases TC1, TC2, and TC3. Test cases TC4, TC5, and TC6 are more challenging, as they have more maxima and TC4 and TC6 also have disconnected fronts. For these reasons, they have design and uncertain vectors of dimension 4, for a total of 8 dimensions. A sensitivity analysis was conducted in order to assess how the several parameters of the solver affect its performance. In particular, coupled analysis for number of function evaluations of inner

TABLE I
TEST FUNCTIONS.

	Function	Parameters
MV1	$f = \sum_{i=1}^n d_i u_i^2$	$\mathbf{d} \in [1, 5]^n$ $\mathbf{u} \in [-5, 3]^n$
MV2	$f = \sum_{i=1}^n (d_i - u_i)^2$	$\mathbf{d} \in [1, 5]^n$ $\mathbf{u} \in [-5, 3]^n$
MV3	$f = \sum_{i=1}^n (5 - d_i)(1 + \cos u_1) + (d_i - 1)(1 + \sin u_i)$	$\mathbf{d} \in [1, 5]^n$ $\mathbf{u} \in [-5, 3]^n$
MV8	$f = \sum_{i=1}^n (2\pi - u_i) \cos(u_i - d_i)$	$\mathbf{d} \in [0, 3]^n$ $\mathbf{u} \in [0, 2\pi]^n$
MV9	$f = \sum_{i=1}^n (d_i - u_i) \cos(-5u_i + 3d_i)$	$\mathbf{d} \in [1, 3]^n$ $\mathbf{u} \in [-\frac{\pi}{2}, \frac{3\pi}{2}]^n$
MV10	$f = \sum_{i=1}^n (d_i + u_i) \times \cos(-u_i(5 d_i + 5) + 3d_i)$	$\mathbf{d} \in [-4, 2\pi]^n$ $\mathbf{u} \in [\pi, 2\pi]^n$
EM1	$f = \sum_{i=1}^n (u_i - 3d_i) \sin u_i + (d_i - 2)^2$	$\mathbf{d} \in [0, 2\pi]^n$ $\mathbf{u} \in [0, 20]^n$

vs. outer loops, total number of agents and number of social agents for MACS ν , and F vs. C_R for IDEA. Note that, as shown in [9], F and C_R of MACS ν do not have a big impact. The sensitivity analysis was run for test case TC4. The results were assessed in terms of success rate of finding the global maximum in the inner loop, indicated as $s.r.(f_i)$, as well as convergence and spreading as per Equations 6 and 7. Results in Tables III - V show that the best success rate, convergence, and spreading are obtained for 20 agents for MACS ν , half of which perform the social actions, and $F = 1$, $C_R = 0.1$ and $200n$ function evaluations for IDEA. The Kriging predictor makes use of a zero-th order polynomial regression function and a Gaussian correlation function. The number of design sites is kept below 20. Three thresholds for the surrogate update have been tested, and they were set to 0.3, 0.5 and 0.9, i.e. 30%, 50% and 90% probability of improvement. The reference solution, i.e. the real front in Figures 2 - 4, was computed by merging the results of 200 runs of the same problem without using the surrogate model and with a number of function evaluations of 10^6 . Tables VI to VIII summarize the performance metrics of the surrogate-based optimization for the six test cases at the several thresholds for the surrogate update. The results are the average performances obtained from the 200 runs needed to achieve a confidence interval of 95% on the success rate being within a $\pm 5\%$ interval containing its estimated value [8]. The column *Inner/Outer* contains the percentage of times that the inner loop was called. The columns M_{conv} and M_{spr} contain the mean value and, in brackets, the standard deviation of M_{conv} and M_{spr} respectively. The columns p_{conv} and p_{spr} contain the success rate of computing a front which convergence and spreading are below the thresholds t_{conv} and t_{spr} , that for this paper were set equal to 5 and 10, respectively.

From Tables VI to VIII it can be seen that the inner loop is called progressively less times as the surrogate update threshold increases, as expected. However, there is a variability among the test cases for the same thresholds. This is due to the fact that certain functions are more or less complicated to approximate by the surrogate model. Note also that there is little difference in the percentage of

TABLE II
TEST CASES.

Test Case	Function	n
TC1	$f_1 = \text{MV1}, f_2 = \text{MV3}$	8
TC2	$f_1 = \text{MV2}, f_2 = \text{MV8}$	8
TC3	$f_1 = \text{MV2}, f_2 = \text{EM1}$	8
TC4	$f_1 = \text{MV8}, f_2 = \text{MV9}$	4
TC5	$f_1 = \text{MV8}, f_2 = \text{EM1}$	4
TC6	$f_1 = \text{MV10}, f_2 = \text{MV9}$	4

TABLE III
SENSITIVITY OF THE ALGORITHM TO THE NUMBER OF AGENTS

$s.r.(f_1, f_2)$ (M_{conv}, M_{spr})		Total agents		
		5	10	20
Social/total	1/3	(100, 46.1) (4.5, 6.8)	(100, 42.5) (3.9, 5.9)	(100, 46.9) (2.6, 4.8)
	1/2	(100, 44.1) (4.6, 6.7)	(100, 41.4) (3.9, 6.1)	(100, 46.3) (2.7, 5.0)
	1	(100, 38.8) (5.1, 7.9)	(100, 40.2) (3.9, 6.1)	(100, 50.7) (2.5, 4.7)

calls to the inner loop when the surrogate update threshold passes from 50% to 90%. This is an indication that the surrogate is able to approximate well the functions, and therefore there are only few points where the probability of improving the surrogate is above 0.9. By and large, the surrogate-based MACS ν produces good results for all the test cases when the surrogate update threshold is 30%, with a computational saving of about 10%. Increasing the update threshold obviously causes the computational saving to be higher, as high as 50% for some cases, because less points are found that have 50% probability of improving the surrogate. This also causes a decrease of the quality of the results in terms of both convergence and spreading. Nevertheless, the solution found is good for four of the six test cases. When the surrogate update threshold is pushed as high as 90%, performances, as well as computational saving, only slightly

TABLE IV
SENSITIVITY OF THE ALGORITHM TO F AND C_R IN THE INNER LOOP

$s.r.(f_1, f_2)$ (M_{conv}, M_{spr})		F			
		0.1	0.5	1	2
C_R	0.1	(99.9, 31.2) (5.5, 7.0)	(100, 41.4) (3.9, 6.1)	(100, 46.6) (3.1, 5.3)	(99.9, 37.8) (4.5, 6.4)
	0.5	(100, 27.3) (6.4, 8.1)	(100, 31.9) (5.6, 7.3)	(100, 36.2) (4.2, 6.3)	(100, 30.5) (5.4, 6.9)
	0.9	(100, 23.8) (8.5, 9.5)	(100, 23.6) (7.6, 8.4)	(100, 27.8) (6.2, 7.6)	(100, 22.7) (8.3, 9.0)

TABLE V
SENSITIVITY OF THE ALGORITHM TO THE NUMBER OF FUNCTION EVALUATIONS

$s.r.(f_1, f_2)$ (M_{conv}, M_{spr})		Outer loop			
		1e5	2e5	3e5	4e5
Inner loop	50n	(100, 19.5) (6.7, 9.4)	(100, 23.1) (6.9, 8.7)	(100, 23.9) (7.7, 9.1)	(100, 25.2) (6.7, 7.8)
	100n	(100, 33.3) (3.5, 7.0)	(100, 35.3) (3.9, 6.5)	(100, 35.7) (4.0, 6.3)	(100, 41.4) (3.9, 6.1)
	100n	(100, 50.6) (2.3, 6.2)	(100, 54.1) (2.5, 5.4)	(100, 59.9) (2.4, 5.2)	(100, 60.0) (2.6, 5.2)

TABLE VI

PERFORMANCE METRICS FOR SURROGATE UPDATE THRESHOLD OF 30%.

Test Case	Inner/Outer	M_{conv}	M_{spr}	p_{conv}	p_{spr}
TC1	99.1%	1.4(0.8)	9.9(5.4)	100%	60%
TC2	92.4%	0.8(0.1)	3.9(1.5)	100%	100%
TC3	87.9%	0.7(0.3)	3.6(1.9)	100%	100%
TC4	89.2%	2.4(1.5)	4.1(1.2)	96%	100%
TC5	92.6%	2.6(2.4)	1.9(1.1)	90%	100%
TC6	87.0%	3.8(0.8)	11.0(4.9)	95%	51%

TABLE VII

PERFORMANCE METRICS FOR SURROGATE UPDATE THRESHOLD OF 50%.

Test Case	Inner/Outer	M_{conv}	M_{spr}	p_{conv}	p_{spr}
TC1	77.6%	2.4(1.2)	16.4(8.9)	98%	26%
TC2	86.6%	0.8(0.1)	5.0(1.9)	100%	98%
TC3	52.1%	3.9(1.6)	12.4(4.4)	77%	30%
TC4	49.6%	3.5(7.3)	6.5(4.3)	86%	87%
TC5	57.4%	12.9(63.1)	8.6(10.1)	44%	75%
TC6	46.1%	5.1(5.0)	25.0(13.7)	72%	12%

TABLE VIII

PERFORMANCE METRICS FOR SURROGATE UPDATE THRESHOLD OF 90%.

Test Case	Inner/Outer	M_{conv}	M_{spr}	p_{conv}	p_{spr}
TC1	75.5%	2.3(1.3)	16.7(7.7)	97%	21%
TC2	86.4%	0.8(0.1)	5.6(2.2)	100%	94%
TC3	51.1%	4.1(1.5)	12.6(4.5)	72%	30%
TC4	46.3%	2.1(1.8)	6.3(3.4)	95%	87%
TC5	53.4%	6.0(3.2)	8.4(6.5)	41%	74%
TC6	41.3%	4.9(2.0)	27.8(11.1)	62%	2%

decrease with respect to the 50% threshold.

The Pareto fronts for the six test cases are shown in Figures 2 to 4 for the three surrogate update threshold of 30%, 50% and 90%. It can be seen that, as mentioned, the surrogate-based optimization algorithm is able to correctly identify the Pareto front for test cases TC2, TC3, TC5, and TC6. Test case TC4 presents a disconnected front, and while the region in the objective space where the real front lies could be identified, the surrogate-based algorithm could not identify the disconnected portions of the front. Test case TC1 proved to be more challenging to solve. Note that some points in the fronts seem to dominate the real solutions. Such points derive from an imperfect approximation by the surrogate.

VI. CONCLUSION

An algorithm for surrogate-based multi-objective worst-case optimization has been presented. The algorithm, called MACS ν , has the peculiarity of performing a number of checks and cross checks to increase the probability to find the global optimum. In this paper, the optimization algorithm is coupled with surrogate modelling. The aim is to create a surrogate model that approximates the mapping between design parameters and function values, so that the expensive inner loop of the min-max algorithm is called only a fraction of the times. The algorithm was tested on six challenging bi-objective test cases made of scalable test functions, presenting multiple maxima and some of them disconnected fronts. An analysis of sensitivity of the parameters of the algorithm was carried out, showing how

several combinations of total vs. social agents, F vs. C_R , and number of function evaluations affected the performance of the algorithm. Moreover, three surrogate update thresholds were tested. The results were good for four out of six test cases, with a good approximation of the real Pareto front obtained with a computational cost of 50% of the exact one. In the case of disconnected fronts, the computed front lays on the real one, though the disconnected portions of the front were not identified. However, this study is limited to the use of the probability of improvement as surrogate update strategy. Other techniques, such as the expected improvement, were proposed in the framework of surrogate-based optimization, and their use in conjunction with MACS ν is under investigation. A real case application of MACS ν is the maximization of Belief function, which finds importance in engineering robust design, for example, as it gives the value of the design budgets for which the design is certainly feasible under uncertainty.

ACKNOWLEDGMENTS

The Kriging predictor used for this work is the one implemented in the DACE Toolbox by Nielsen, Lophaven and Sondergaard, and freely available at <http://www2.imm.dtu.dk/~hbni/dace/>. The authors would like to thank the anonymous reviewers for helping to improve the quality of the paper. This work is partially supported through an ESA/NPI grant ‘Evidence-based Robust Design Optimization’.

REFERENCES

- [1] M. Vasile, E. Minisci and Q. Wijnands, “Approximated Computation of Belief Functions for Robust Design Optimization,” *53rd AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference*, Apr. 2012.
- [2] M. Bhattacharya, “Surrogate based Evolutionary Algorithm for Design Optimization,” *World Academy of Science, Engineering and Technology* vol. 1, 2007.
- [3] Y. Jin, “Surrogate-Assisted Evolutionary Computation: Recent Advances and Future Challenges,” *Swarm and Evolutionary Computation* vol. 1, pp 61-70, 2011.
- [4] A. Zhou and Q. Zhang, “A Surrogate-Assisted Evolutionary Algorithm for Minimax Optimization,” *2010 IEEE Conference on Evolutionary Computation (CEC 2010)*, Jul. 2010.
- [5] Y.-S. Ong, P. B. Nair, K. Y. Lum, “Max-Min Surrogate-Assisted Evolutionary Algorithm for Robust Design,” *IEEE Trans. Evol. Comp.* vol. 10, pp 392-404, 2006.
- [6] J. Marzat, E. Walker and H. Piet-Lahanier, “Worst-case Global Optimization of Black-Box Functions through Kriging and Relaxation,” *J. Global Optim.* vol. 55, pp. 707-727, 2013.
- [7] G. Shafer, *A Mathematical Theory of Evidence*. Princeton University Press, Princeton, NJ, 1976.
- [8] M. Vasile, E. Minisci and M. Locatelli, “An Inflationary Differential Evolution Algorithm for Space Trajectory Optimization,” *IEEE Trans. Evol. Comp.* vol. 15, pp. 267-281, Apr. 2011.
- [9] F. Zuiani and M. Vasile, “Multi Agent Collaborative Search Based on Tchebycheff Decomposition,” *Computational Optimization and Applications*, March 2013, DOI 10.1007/s10589-013-9552-9.
- [10] D. R. Jones, “A Taxonomy of Global Optimization Methods Based on Response Surface,” *J. Global Optim.* vol. 21, pp. 345-383, 2001.
- [11] A. I. J. Forrester and A. J. Keane, “Recent Advances in Surrogate-based Optimization,” *Prog Aerosp Sci.* vol. 45, pp. 50-79, 2009.
- [12] A. J. Keane, “Statistical Improvement Criteria for Use in Multiobjective Design Optimization,” *AIAA Journal*, vol. 44, no. 4, pp. 879-891, 2006.

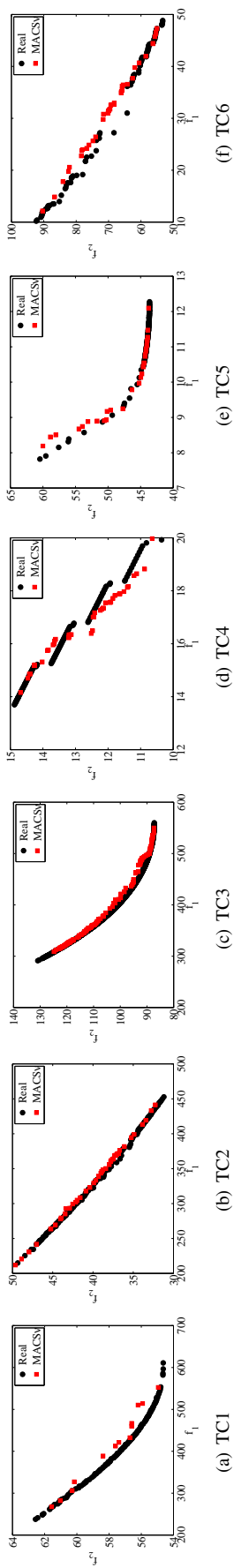


Fig. 2
PARETO FRONTS OF THE TEST CASES WITH SURROGATE UPDATE THRESHOLD OF 30%.

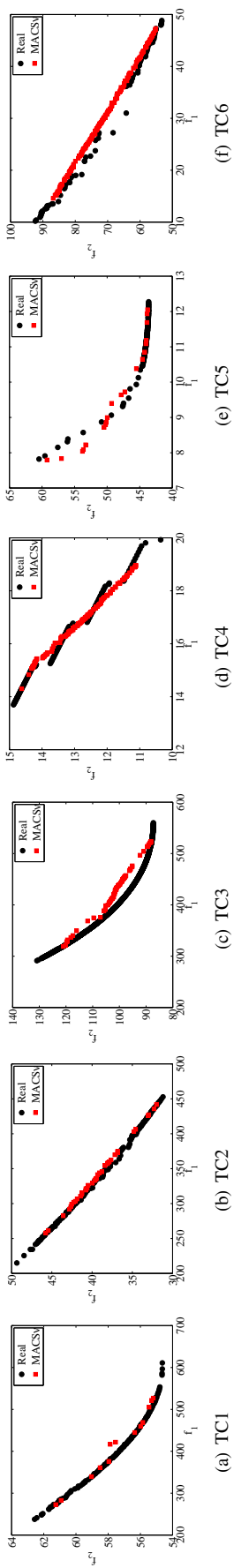


Fig. 3
PARETO FRONTS OF THE TEST CASES WITH SURROGATE UPDATE THRESHOLD OF 50%.

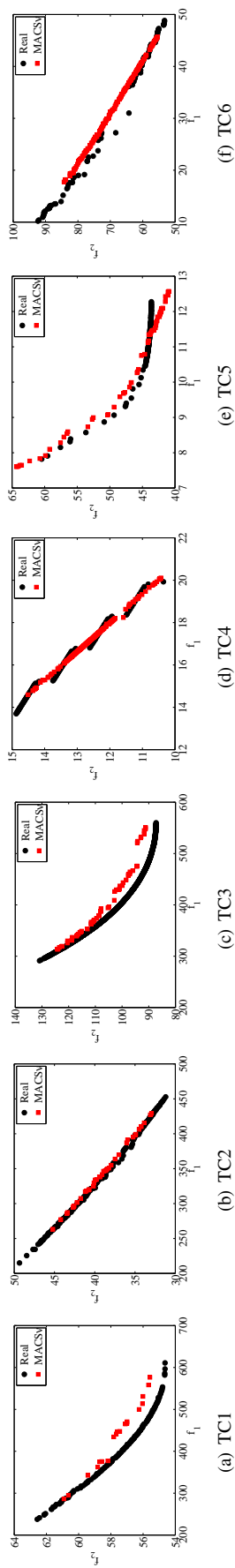


Fig. 4
PARETO FRONTS OF THE TEST CASES WITH SURROGATE UPDATE THRESHOLD OF 90%.