ANALYSIS OF TWO ALGORITHMS FOR MULTI-OBJECTIVE MIN-MAX OPTIMIZATION

Simone Alicino
Mechanical and Aerospace Engineering
University of Strathclyde, Glasgow, UK
simone.alicino@strath.ac.uk

Massimiliano Vasile
Mechanical and Aerospace Engineering
University of Strathclyde, Glasgow, UK
massimiliano.vasile@strath.ac.uk

Abstract This paper presents two memetic algorithms to solve multi-objective min-max problems, such as the ones that arise in evidence-based robust optimization. Indeed, 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. A number of problems, composed of functions whose uncertain space is modelled by means of Evidence Theory, and presenting multiple local maxima as well as concave, convex, and disconnected fronts, are used to test the performance of the proposed algorithms.

Keywords: Multi-objective optimization, worst-case scenario design, evidence-based robust optimization.

1. Introduction

Worst-case scenario problems arise whenever a performance index, or cost function, has to be optimal with respect to a design vector $\mathbf{d}$, and at the same time robust against an uncertain vector $\mathbf{u}$. This class of problems is common in several fields, such as game theory, decision making, robust control, risk analysis, and robust design. For instance, the lower expectation in the realization of the value of a particular performance index for a model of a system can be defined as the degree of belief that

Algorithm 1 Min-max optimization via restoration

1: Initialize archive $A_u = \{\mathbf{u}_1\}$, and set $i = 1$
2: **while** the stopping condition is not met **do**
3: Compute $\mathbf{d}_i = \arg\min_{\mathbf{d}\in D} \left\{ \max_{\mathbf{u}\in A_u} f(\mathbf{d}, \mathbf{u}) \right\}$
4: Compute $\mathbf{u}_{i+1} = \arg\max_{\mathbf{u}\in U} f(\mathbf{d}_i, \mathbf{u})$
5: Add $\mathbf{u}_{i+1}$ to the archive A_u
6: $i \leftarrow i + 1$
7: **end while**
8: Return $\{\mathbf{d}_i, \mathbf{u}_{i+1}\}$.

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 an uncertain quantity is available. Stochastic variables and associated probability are replaced by a multivalued mapping from a collection of subsets of an uncertain space U into a lower expectation (Belief function in the case of Evidence Theory). The main drawback of the use of multivalued mappings is that the computation of the lower expectation, i.e. the Belief, has a complexity that is exponential with the number of uncertain variables. Recently, some strategies were proposed in [1] 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. Several methods have been proposed to address single-objective min-max problems, especially using evolutionary approaches [2, 3], and metamodels [4–6]. For the multi-objective case, a gradient-based approach is presented in [7]. An interesting approach is based on the procedure proposed in [6, 8] for single-objective problems, and exploited in [9] for interval multi-objective linear programming. Such procedure is based on an iterative minimization over the design space and subsequent restoration of the global maximum over the uncertain space as shown in Algorithm 1. The stopping condition can be the achievement of a desired accuracy, or a maximum number of function evaluations, for example. In this paper we present a multi-objective version of Algorithm 1 implemented in an algorithm called

MACSminmax. MACSminmax, employs MACS2 and IDEA at steps 3 and 4, respectively. Another algorithm, MACSν, is presented in this paper and compared to MACSminmax. MACSν is a variant of MACS2 containing heuristics to deal with min-max problems. The paper starts with a brief introduction to Evidence Theory and its use in the context of robust design optimization in Section 2. Section 3 introduces the two memetic algorithms, MACSminmax and MACSν. Section 4 finally presents the results on some test cases.

2. Evidence-Based Robust Design Optimization

Evidence Theory [10] 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 quantified by using a mass function generally known as Basic Probability Assignment (*bpa*). All the intervals form the so-called frame of discernment Θ, which is a set of mutually exclusive elementary propositions. 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, defined as follows:

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

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

$$A = \{\mathbf{u} \in U \mid f(\mathbf{u}) \leq \nu\} \tag{2}$$

where f is the outcome of the system model and the threshold ν is the desired value of a design budget. 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. This introduces discontinuities in the search space, making the problem more difficult to solve.

An engineering system to be optimized can be modelled as a function $f : D \times U \subseteq \Re^{m+n} \rightarrow \Re$. 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 $\boldsymbol{\nu}_{\max}$ 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:

$$\boldsymbol{\nu}_{\max} = \min_{\mathbf{d} \in D} \mathbf{F} = \min_{\mathbf{d} \in D} [\max_{\mathbf{u} \in \bar{U}} f_1(\mathbf{d}, \mathbf{u}), \ldots, \max_{\mathbf{u} \in \bar{U}} f_q(\mathbf{d}, \mathbf{u})]^T \tag{3}$$

Problem in 3 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:

$$\mathbf{x}_U = \frac{\left(\mathbf{b}^u_{U,i} - \mathbf{b}^l_{U,i}\right)}{\left(\mathbf{b}^u_{\bar{U},i} - \mathbf{b}^l_{\bar{U},i}\right)} \mathbf{x}_{\bar{U},i} + \mathbf{b}^l_{U,i} - \frac{\left(\mathbf{b}^u_{U,i} - \mathbf{b}^l_{U,i}\right)}{\left(\mathbf{b}^u_{\bar{U},i} - \mathbf{b}^l_{\bar{U},i}\right)} \mathbf{b}^l_{\bar{U},i} \tag{4}$$

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

3. Multi-Objective Min-Max Memetic Optimization

Problem (3) 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 MACS2. IDEA [11] is a population-based memetic algorithm for single-objective optimization. It hybridizes Differential Evolution and Monotonic Basin Hopping in order to simultaneously improve local convergence and avoid stagnation. MACS2 [12] 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. Individualistic actions perform a sampling of the search space in a neighborhood 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.

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 $\tilde{\mathbf{u}}_1$ and $\tilde{\mathbf{u}}_2$. 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.

3.1 MACSν

MACSν (Algorithm 2) is the min-max variant of MACS2. It endows MACS2 with special heuristics to increase the probability of finding the global maxima in $\bar{U}$. More in detail, a Cross-check (lines 7, 18, and 28) compares the values of the objective functions for a newly generated design vector in the trial populations P_t (line 7) and P_s (line 18) against the function values of a solution already archived in A (indicated with subscript *arch* in Algorithm 2). In addition, 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. After the cross-check, a Min-Max Selection (lines 11 and 22) compares the population P with the new candidate populations P_t (line 11) and P_s (line 22) and selects the design vectors to attribute to the next generation according to the following rule: 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 higher (resp. lower) value of the objective function; if both $\mathbf{d}$ and $\mathbf{u}$ are different, the new vectors will replace the old ones. At the end of the algorithm, and at the last iteration, a Validation (line 24) mitigates the possibility that the cross-check operators assign the same incorrect $\mathbf{u}$ to all $\mathbf{d}$ vectors in the population and archive. This is done by starting from the minimum value of the first objective in the archived Pareto front, and performing a global search in the uncertain space. If the new uncertain vector gives a higher value of the function, then it replaces the old one. This operation is repeated for the elements in the archived Pareto front until there is no more variation in their value.

Algorithm 2 MACSν

Initialize population P, archive $A = P$, $n_{feval} = 0$, $\epsilon = 0.7$, $\delta = 10^{-6}$
while $n_{feval} < n_{feval,max}$ **do**
  Run individualistic moves and generate trial population P_t
  for all $\mathbf{d} \in P_t$ **do**
    for all $\mathbf{d}_{arch} \in A$ **do**
      if $\mathbf{d} \succ \mathbf{d}_{arch}$ **then**
        Cross-check(P_t, A)
      end if
    end for
  end for
  Min-Max Selection(P, P_t)
  Update P and A
  $Z \leftarrow \|\mathbf{F}_{arch}^{max} - \mathbf{F}_{arch}^{min}\|$
  Run social moves and generate candidate population P_s
  for all $\mathbf{d} \in P_s$ **do**
    for all $\mathbf{d}_{arch} \in A$ **do**
      if $\mathbf{d} \succ \mathbf{d}_{arch}$ **or** $\|\mathbf{F}(\mathbf{d}) - \mathbf{F}(\mathbf{d}_{arch})\| > \epsilon Z$ **then**
        Cross-check(P_s, A)
      end if
    end for
  end for
  Min-Max Selection(P, P_s)
  Update P and A
  Validation(A)
  for all $\mathbf{d} \in P$ **do**
    for all $\mathbf{d}_{arch} \in A$ **do**
      if $\mathbf{d} \succ \mathbf{d}_{arch}$ or $\mathbf{d} \prec \mathbf{d}_{arch}$ **then**
        Cross-check(P, A)
      else if $\|\mathbf{F}(\mathbf{d}) - \mathbf{F}(\mathbf{d}_{arch})\| < \delta$ **then**
        Replace $\mathbf{u} \in P$ with $\mathbf{u} \in A$
      end if
    end for
  end for
end while

3.2 MACSminmax

MACSminmax (Algorithm 3) is a min-max memetic algorithm inspired by the procedure of Algorithm 1. This is the main difference with MACSν. In MACSminmax, for each agent of the minimization the best function value is computed with respect to an archive A_u of candidate

Algorithm 3 MACSminmax

1: Initialize archive $A_u = \{\mathbf{u}_1\}$, $n_{feval} = 0$
2: **while** $n_{feval} < n_{feval,max}$ **do**
3: Run MACS2 to compute $\mathbf{d}_{min} = \arg\min_{\mathbf{d}\in D} \max_{\mathbf{u}\in A_u} f(\mathbf{d}, \mathbf{u})$ and associated $\mathbf{f_d}$
4: Add $\mathbf{d}_{min}$ to the archive A_d
5: **for all** $\mathbf{d}_{min} \in A_d$ **do**
6: **for all** $l \in \{1, \ldots, q\}$ **do**
7: Run IDEA to compute $\mathbf{u}^l_{max} = \arg\max_{\mathbf{u}\in\bar{U}} f^l(\mathbf{d}_{min}, \mathbf{u})$ and associated $f^l_{\mathbf{u}}$
8: **if** $f^l_{\mathbf{u}} > f^l_{\mathbf{d}}$ **then**
9: Add $\mathbf{u}^l_{max}$ to the archive A_u
10: **else**
11: Evaluate function to find $\mathbf{u}^l_{max} = \arg\max_{\mathbf{u}\in A_u} f^l(\mathbf{d}_{min}, \mathbf{u})$
12: **end if**
13: **end for**
14: **end for**
15: **end while**
16: **for all** $\mathbf{d}_{min} \in A_d$ **do**
17: **for all** $l \in \{1, \ldots, q\}$ **do**
18: Run local search to refine $\mathbf{u}^l_{max} \in A_u$ associated to $\mathbf{d}_{min}$
19: **end for**
20: **end for**
21: Return non-dominated $\mathbf{d}_{min}$ and associated $\mathbf{u}^l_{max}$

uncertain vectors (line 3). The archive A_u is composed of the results of a global maximization or a simple function evaluation, depending on which one of the two gives the higher function value (as explained above, it is not guaranteed that the maximization finds the global maximum), for each design vector contained in another archive A_d of candidate solutions (lines 5 to 14). Thus, each element in the archive A_u corresponds to an element in the archive A_d. This is so if the global maxima change for different design vectors. If they do not change, the archive A_u is composed of only one element. At the end of the main loop, a local search is run for each element of the archive A_d in order to refine the accuracy of the elements in the archive A_u. Finally, because the archive A_d is filled with batches of solutions given in output by MACS2, the solutions are non-dominated only inside each batch. Therefore a further dominance check is necessary to find the non-dominated solutions among the batches.

Interesting is a comparison between MACSminmax and MACSν. In MACSν a maximization is run for every agent of the minimization, whereas in MACSminmax each agent of the minimization is cross-checked with the archive of candidate uncertain vectors through a function evaluation. However, it is worth noting that the evaluation, in MACSminmax, of each $\mathbf{d}$ against an archive A_u of candidate uncertain vectors, as well as the update of A_u for each element of an archive A_d of candidate design vectors, is equivalent to the cross-checks implemented in MACSν. Furthermore, the local search in MACSminmax after the main loop is similar to the validation procedure in MACSν, where a global search is run starting from the extrema of the Pareto front. Finally, in terms of balance between exploration (social moves) and exploitation (individualistic moves) of the search space, both MACSν and MACSminmax employ the same search algorithms, MACS2 and IDEA, therefore they are equivalent so long as the parameters (population, F, C_R) are set to the same values.

4. Test Cases

MACSν and MACSminmax were tested on the six bi-objective and one tri-objective test cases reported in Table 1, where n is the dimension of the design vector $\mathbf{d}$, as well as the uncertain vector $\mathbf{u}$ – therefore the total dimension of the test cases is $2n$ – and $n_{feval,max}$ is the maximum number of function evaluations, i.e. the termination condition for the algorithms. The test cases are composed of the functions in Table 2. The functions are easily scalable and 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. Note also that the test cases present several types of Pareto fronts, convex, concave, linear, and disconnected. The uncertain vector $\mathbf{u}$ is assigned the *bpa* structure reported in Table 3. The uncertain intervals present holes and overlappings, that introduce discontinuities in the uncertain space. The reference solution, i.e. the real front in Figures 1, was computed by merging the results of 200 runs of the same problems solved by means of MACSν with the results of 200 runs of MACSminmax.

From a sensitivity analysis on total number of agents (5, 10, 20) vs. subset of social agents (1/3, 1/2, 1), and F (0.1, 0.5, 1, 2) vs. C_R (0.1, 0.5, 0.9) for MACS2 and IDEA resulted that the best settings were: $200n$ function evaluations for both MACS2 and IDEA, for 10 agents for MACS2, half of which perform the social actions, 5 agents for IDEA, and $F = 1$ and $C_R = 0.1$ for both MACS2 and IDEA. The sensitivity

Table 1. Test cases.

Test Case	Functions	$\mathbf{d}$	n	$n_{feval,max}$
TC1	f_1 = MV1, f_2 = MV3	$[1,5]^n$	2	2E5
TC2	f_1 = MV2, f_2 = MV8	$[0,3]^n$	8	1E6
TC3	f_1 = MV2, f_2 = EM1	$[1,5]^n$	8	1E6
TC4	f_1 = MV8, f_2 = MV9	$[1,3]^n$	2	4E5
TC5	f_1 = MV8, f_2 = EM1	$[1,5]^n$	4	1E6
TC6	f_1 = MV10, f_2 = MV9	$[-4,2\pi]^n$	1	1E5
TC7	f_1 = MV2, f_2 = MV8, f_3 = EM1	$[1,5]^n$	4	1E6

Table 2. Test functions.

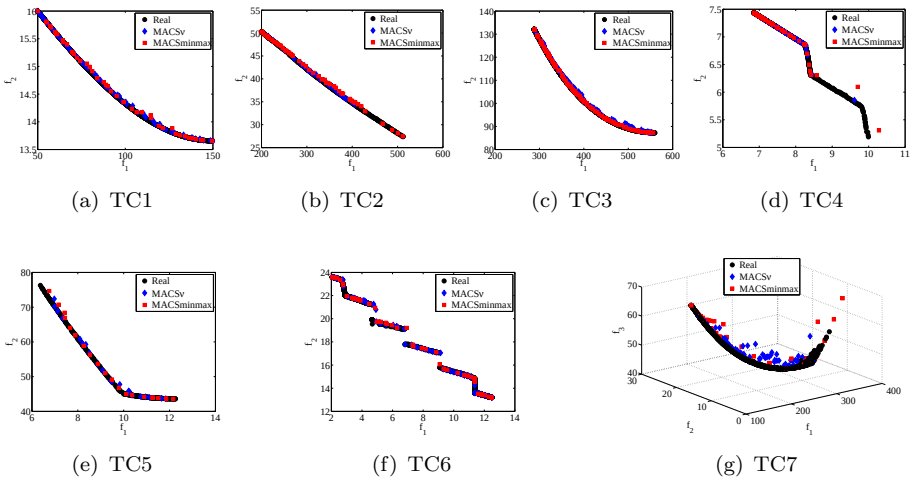
ID	Function
MV1	$f = \sum_{i=1}^{n} d_i u_i^2$
MV2	$f = \sum_{i=1}^{n} (d_i - u_i)^2$
MV3	$f = \sum_{i=1}^{n} (5 - d_i)(1 + \cos u_1) + (d_i - 1)(1 + \sin u_i)$
MV8	$f = \sum_{i=1}^{n} (2\pi - u_i)\cos(u_i - d_i)$
MV9	$f = \sum_{i=1}^{n} (d_i - u_i)\cos(-5u_i + 3d_i)$
MV10	$f = \sum_{i=1}^{n} (d_i + u_i)\cos(-u_i(5\lvert d\rvert + 5) + 3d_i)$
EM1	$f = \sum_{i=1}^{n} (u_i - 3d_i)\sin u_i + (d_i - 2)^2$

Table 3. *bpa* structure of the uncertain variables.

MV1, MV2, MV3	Interval	[-5 -4]	[-3 0]	[-1 3]
	bpa	0.1	0.25	0.65
MV8	Interval	[0 1]	[2 4]	[3 2π]
	bpa	0.1	0.25	0.65
MV9	Interval	[-π/2 -π/6]	[0 π]	[3π/4 3π/2]
	bpa	0.1	0.4	0.5
MV10	Interval	[π 4]	[5 6]	[5.5 2π]
	bpa	0.1	0.25	0.65
EM1	Interval	[0 5]	[7 14]	[12 20]
	bpa	0.1	0.5	0.4

analyses were run for test case TC4 with a total of 2E5 function evaluations, and the results assessed in terms of success rate of finding the global maximum, as well as convergence M_{conv} and spreading M_{spr} as per definition in [13]. The same settings were used in all the test cases.

Table 4 summarizes the success rates of finding the global maxima, as well as convergence and spreading of MACSminmax in comparison to MACSν. 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 [11]. Columns $maxf_1$, $maxf_2$ and $maxf_3$ contain the maximization success rates computed with an accuracy of 10^{-4} with respect to the actual maxima, columns M_{conv} and M_{spr} contain the mean value of M_{conv} and M_{spr} respectively, and columns p_{conv}/t_{conv} and p_{spr}/t_{spr} contain the success rate of computing a front which convergence and spreading are below the thresholds t_{conv} and t_{spr} also contained in the columns after the '/' symbol. MACSminmax attains performances similar to MACSν, with excellent success rates for almost all the test cases. For TC2, TC3 and TC7, MACSminmax provides a significantly better spreading (2.0, 0.3, and 2.1) than MACSν (16.1, 7.5, and 9.3). Note also that TC2 and TC3 are the test cases with the higher dimension, 16, whereas TC7 has the highest number of objectives, 3. Moreover, for TC5 MACSminmax has a significantly higher success rate for the maximization of the second objective (87.6% against 54.1%): in function EM1 the global maximum has a jump for a certain value of $\mathbf{d}$. This makes the global maximum been tracked more effectively with the global search implemented in MACSminmax than with the local search of MACSν. However, for TC5 average converge and spreading computed by MACSminmax, and their success rates, are worse than for MACSν. MACSν also performs better at finding a front for TC6 which spreading is below a threshold equal to 2. In conclusion, MACSminmax has equal or better capability in the maximization in the uncertain space, and also in terms of convergence and spreading, than MACSν, which in turns performed slightly better in two cases. On one hand, such rather equivalent performances of the two algorithms can be explained by the fact that they have equivalent balance between exploration and exploitation, as explained in subsection 3.2. On the other hand, the better performance of MACSminmax on some of the test cases can be due to more effective archiving, cross-check and validation mechanisms, which are the only aspects that differentiate MACSν and MACSminmax. The Pareto fronts for the seven test cases are shown in Figure 1. As a comparison, the fronts computed by means of MACSν and of MACSminmax are displayed. One can see that MACSminmax performs as well as MACSν at identifying the true Pareto front for all the test cases. In addition, it is worth noting that TC4 presents a deceptive front, as the bottom-right portion of it has a multitude of dominated fronts above it. This resulted to be a very difficult part for both MACSν and MACSminmax to identify.



(a) TC1 (b) TC2 (c) TC3 (d) TC4

(e) TC5 (f) TC6 (g) TC7

Figure 1. Pareto fronts of the test cases.

Table 4. Results: comparison between MACSν and MACSminmax.

Test Case	Algorithm	$max f_1$	$max f_2$	$max f_3$	M_{conv}	M_{spr}	p_{conv} / t_{conv}	p_{spr} / t_{spr}
TC1	MACSν	100%	100%	-	0.2	1.7	100 / 0.5	79 / 2
	MACSminmax	100%	100%	-	0.2	1.3	100 / 0.5	100 / 2
TC2	MACSν	100%	65%	-	0.5	16.1	100 / 1	0 / 2
	MACSminmax	100%	60%	-	0.6	2.0	100 / 1	64 / 2
TC3	MACSν	100%	100%	-	0.6	7.5	46 / 0.5	3 / 2
	MACSminmax	100%	100%	-	0.1	0.3	100 / 0.5	100 / 2
TC4	MACSν	100%	91.3%	-	0.3	0.9	83 / 0.5	97 / 2
	MACSminmax	100%	85.7%	-	0.4	1.0	77 / 0.5	91 / 2
TC5	MACSν	98.6%	54.1%	-	1.2	5.8	48 / 1	60 / 6
	MACSminmax	92.8%	87.6%	-	2.7	8.0	24 / 1	42 / 6
TC6	MACSν	100%	100%	-	0.3	1.2	95 / 0.5	97 / 2
	MACSminmax	100%	100%	-	0.3	2.0	91 / 0.5	63 / 2
TC7	MACSν	100%	100%	95.3%	5.0	9.3	50 / 5	8 / 5
	MACSminmax	100%	100%	98.3%	4.6	2.1	66 / 5	100 / 5

5. Conclusions

Two multi-objective min-max memetic algorithms, MACSminmax and MACSν have been presented and compared in this paper. MACSν is a variant of MACS2 endowed with cross-checks, and selection and validation mechanisms to properly maximize the subproblem. MACSminmax makes use of an iterative restoration of the global maxima in the uncertain space. Despite the different procedures, the two strategies imple-

ment similar cross-checks. The two algorithms have been tested on seven scalable test cases that present several types of Pareto fronts. Results show that both MACSminmax and MACSν are able to achieve similar very good performances, in terms of finding the global maxima in the uncertain space and the true Pareto front. However, MACSminmax performed significanly better in terms of spreading in the two test cases with the highest dimension and the one with three objectives. Multi-objective min-max optimization algorithms find applicability to worst-case scenario problems, such as evidence-based robust engineering design.

References

[1] M. Vasile, E. Minisci, and Q. Wijnands. Approximated Computation of Belief Functions for Robust Design Optimization. In *53rd AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference*, Honolulu, USA, 2012.

[2] A.M. Cramer, S.D. Sudhoff, and E.L. Zivi. Evolutionary Algorithms for Minimax Problems in Robust Design. *IEEE Trans. Evol. Comp.*, 13(2):444–453, 2009.

[3] R.I. Lung and D. Dumitrescu. A New Evolutionary Approach to Minimax Problems. In *IEEE Congress on Evolutionary Computation*, New Orleans, USA, 2011.

[4] A. Zhou and Q. Zhang. A Surrogate-Assisted Evolutionary Algorithm for Minimax Optimization. In *IEEE Congress on Evolutionary Computation*, Barcelona, Spain, 2010.

[5] Y.-S. Ong, P.B. Nair, and K.Y. Lum. Max-Min Surrogate-Assisted Evolutionary Algorithm for Robust Design. *IEEE Trans. Evol. Comp.*, 10:392–404, 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.*, 55:707–727, 2013.

[7] S. Azarm and H. Eschenauer. A Minimax Reduction Method for Multi-Objective Decomposition-Based Design Optimization. *Structural Optimization*, 6:94–98, 1993.

[8] K. Shimizu and E. Aiyoshi. Necessary Conditions for Min-Max Problems and Algorithms by a Relaxation Procedure. *IEEE Trans. Automat. Contr.*, 25(1):62–66, 1980.

[9] S. Rivaz and M.A. Yaghoobi. Minimax Regret Solution to Multiobjective Linear Programming Problems with Interval Objective Functions Coefficients. *Cent. Eur. J. Oper. Res.*, 21:625–649, 2013

[10] G. Shafer *A Mathematical Theory of Evidence*. Princeton University Press, 1976.

[11] M. Vasile, E. Minisci, and M. Locatelli. An Inflationary Differential Evolution Algorithm for Space Trajectory Optimization. *IEEE Trans. Evol. Comp.*, 15:267–281, 2011.

[12] F. Zuiani and M. Vasile. Multi Agent Collaborative Search Based on Tchebycheff Decomposition. *Comput. Optim. Appl.*, 56(1):189–208, 2013.

[13] M. Vasile and F. Zuiani. Multi-agent Collaborative Search: An Agent-based Memetic Multi-Objective Optimization Algorithm Applied to Space Trajectory Design. *Proc. Inst. Mech. Eng. G J. Aerosp. Eng.* 225:1211–1227, 2011.