# Multi-agent collaborative search: an agent-based memetic multi-objective optimization algorithm applied to space trajectory design

M Vasile and F Zuiani

*Proceedings of the Institution of Mechanical Engineers, Part G: Journal of Aerospace Engineering* 2011 225: 1211
originally published online 5 September 2011
DOI: 10.1177/0954410011410274

# Multi-agent collaborative search: an agent-based memetic multi-objective optimization algorithm applied to space trajectory design

**M Vasile*** and **F Zuiani**
Department of Mechanical and Aerospace Engineering, University of Strathclyde, Glasgow, UK

*The manuscript was received on 2 November 2010 and was accepted after revision for publication on 21 April 2011.*

DOI: 10.1177/0954410011410274

**Abstract:** This article presents an algorithm for multi-objective optimization that blends together a number of heuristics. A population of agents combines heuristics that aim at exploring the search space both globally and in a neighbourhood of each agent. These heuristics are complemented with a combination of a local and global archive. The novel agent-based algorithm is tested at first on a set of standard problems and then on three specific problems in space trajectory design. Its performance is compared against a number of state-of-the-art multi-objective optimization algorithms that use the Pareto dominance as selection criterion: non-dominated sorting genetic algorithm (NSGA-II), Pareto archived evolution strategy (PAES), multiple objective particle swarm optimization (MOPSO), and multiple trajectory search (MTS). The results demonstrate that the agent-based search can identify parts of the Pareto set that the other algorithms were not able to capture. Furthermore, convergence is statistically better although the variance of the results is in some cases higher.

**Keywords:** multi-objective optimization, trajectory optimization, memetic algorithms, multiagent systems

## 1 INTRODUCTION

The design of a space mission steps through different phases of increasing complexity. In the first phase, a trade-off analysis of several options is required. The trade-off analysis compares and contrasts design solutions according to different criteria and aims at selecting one or two options that satisfy mission requirements. In mathematical terms, the problem can be formulated as a multi-objective optimization problem.

As part of the trade-off analysis, multiple transfer trajectories to the destination need to be designed. Each transfer should be optimal with respect to a number of criteria. The solution of the associated multi-objective optimization problem, has been addressed, by many authors, with evolutionary techniques. Coverstone-Caroll *et al.* [**1**] proposed the use of multi-objective genetic algorithms for the optimal design of low-thrust trajectories. Dachwald *et al.* [**2**] proposed the combination of a neurocontroller and of a multi-objective evolutionary algorithm for the design of low-thrust trajectories. In 2005, a study by Lee *et al.* [**3**] proposed the use of a Lyapunov controller with a multi-objective evolutionary algorithm for the design of low-thrust spirals. More recently, Schütze *et al.* [**4**] proposed some innovative techniques to solve multi-objective optimization problems for multi-gravity low-thrust trajectories. Two of the interesting aspects of the work of Schütze *et al.* [**4**] are the archiving of $\epsilon$- and $\Delta$-approximated solutions, to the known best Pareto front, and the deterministic pre-pruning of the search space [**5**]. In 2009, Dellnitz *et al.* [**6**] proposed the use of multi-objective subdivision techniques for the design of low-thrust transfers to the halo orbits around the $L_2$ libration point in the Earth–Moon system. Minisci

**Corresponding author: Department of Mechanical and Aerospace Engineering, University of Strathclyde, James Weir Building, 75 Montrose Street, Glasgow G1 1XJ, UK.*
*email: massimiliano.vasile@strath.ac.uk*

*et al.* [7] presented an interesting comparison between an EDA-based algorithm, called MOPED, and non-dominated sorting genetic algorithm (NSGA-II) on some constrained and unconstrained multi-impulse orbital transfer problems.

In this article, a hybrid population-based approach that blends a number of heuristics is proposed. In particular, the search for Pareto optimal solutions is carried out globally by a population of agents implementing classical social heuristics and more locally by a subpopulation implementing a number of individualistic actions. The reconstruction of the set of Pareto optimal solutions is handled through two archives: a local and a global one.

The individualistic actions presented in this article are devised to allow each agent to independently converge to the Pareto optimal set, thus creating its own partial representation of the Pareto front. Therefore, they can be regarded as memetic mechanisms associated to a single individual. It will be shown that individualistic actions significantly improve the performance of the algorithm.

The algorithm proposed in this article is an extension of the multi-agent collaborative search (MACS), initially proposed in references [**8**, **9**], to the solution of multi-objective optimization problems. Such an extension required the modification of the selection criterion, for both global and local moves, to handle Pareto dominance and the inclusion of new heuristics to allow the agents to move toward and along the Pareto front. As part of these new heuristics, this article introduces a dual archiving mechanism for the management of locally and globally Pareto optimal solutions and an attraction mechanism that improves the convergence of the population.

The new algorithm is here applied to a set of known standard test cases and to three space mission design problems. The space mission design cases in this article consider spacecraft equipped with a chemical engine and performing a multi-impulse transfer. Although these cases are different from some of the above-mentioned examples, that consider a low-thrust propulsion system, nonetheless the size and complexity of the search space is comparable. Furthermore, it provides a first test benchmark for multi-impulsive problems that have been extensively studied in the single objective case but for which only few comparative studies exist in the multi-objective case [7].

This article is organized as follows: section 2 contains the general formulation of the problem. Section 3 starts with a general introduction to the multi-agent collaborative search algorithm and heuristics before going into some of the implementation details. Section 4 contains a set of comparative tests that demonstrates the effectiveness of the heuristics implemented in MACS. The section briefly introduces the algorithms against which MACS is compared and the two test benchmarks that are used in the numerical experiments. It then defines the performance metrics and ends with the results of the comparison.

## 2 PROBLEM FORMULATION

A general problem in multi-objective optimization is to find the feasible set of solutions that satisfies the following problem

$$\min_{\mathbf{x}\in D} \mathbf{f}(\mathbf{x}) \tag{1}$$

where $D$ is a hyperrectangle defined as $D = \left\{x_j \mid x_j \in [b_j^l \; b_j^u] \subseteq \mathbb{R},\ j = 1, \ldots, n\right\}$ and $\mathbf{f}$ the vector function

$$\mathbf{f} : D \to \mathbb{R}^m, \qquad \mathbf{f}(\mathbf{x}) = [f_1(\mathbf{x}), f_2(\mathbf{x}), \ldots, f_m(\mathbf{x})]^T \tag{2}$$

The optimality of a particular solution is defined through the concept of dominance: with reference to problem (1), a vector $\mathbf{y} \in D$ is dominated by a vector $\mathbf{x} \in D$ if $f_j(\mathbf{x}) < f_j(\mathbf{y})$ for all $j = 1, \ldots, m$. The relation $\mathbf{x} \prec \mathbf{y}$ states that $\mathbf{x}$ dominates $\mathbf{y}$.

Starting from the concept of dominance, it is possible to associate, to each solution in a set, the scalar dominance index

$$I_d(\mathbf{x}_j) = |\{i \mid i \wedge j \in N_p \wedge \mathbf{x}_i \prec \mathbf{x}_j\}| \tag{3}$$

where the symbol $|.|$ is used to denote the cardinality of a set and $N_p$ is the set of the indices of all the solutions. All non-dominated and feasible solutions form the set

$$X = \{\mathbf{x} \in D \mid I_d(\mathbf{x}) = 0\} \tag{4}$$

Therefore, the solution of problem (1) translates into finding the elements of $X$. If $X$ is made of a collection of compact sets of finite measure in $\mathbb{R}^n$, then once an element of $X$ is identified, it makes sense to explore its neighbourhood to look for other elements of $X$. On the other hand, the set of non-dominated solutions can be disconnected and its elements can form islands in $D$. Hence, restarting the search process in unexplored regions of $D$ can increase the collection of elements of $X$.

The set $X$ is the Pareto set and the corresponding image in criteria space is the Pareto front. It is clear that in $D$, there can be more than one $X_l$ containing solutions that are locally non-dominated, or locally Pareto optimal. The interest is, however, to find the set $X_g$ that contains globally non-dominated, or globally Pareto optimal, solutions.

## 3 MULTIAGENT COLLABORATIVE SEARCH

The key motivation behind the development of multiagent collaborative search was to combine local and global search in a coordinated way such that local convergence is improved while retaining global exploration [**9**]. This combination of local and global search is achieved by endowing a set of agents with a repertoire of actions producing either the sampling of the whole search space or the exploration of a neighbourhood of each agent. More precisely, in the following, global exploration moves will be called collaborative actions while local moves will be called individualistic actions. Note that not all the individualistic actions, described in this article, aim at exploring a neighbourhood of each agent, though. The algorithm presented in this article is a modification of MACS to tackle multi-objective optimization problems. In this section, the key heuristics underneath MACS will be described together with their modification to handle problem (1) and reconstruct $X$.

### 3.1 General algorithm description

A population $P_0$ of $n_{\text{pop}}$ virtual agents, one for each solution vector $\mathbf{x}_i$, with $i=1,\ldots, n_{\text{pop}}$, is deployed in $D$. The population evolves through a number of generations. At every generation $k$, the dominance index (3) of each agent $\mathbf{x}_{i,k}$ in the population $P_k$ is evaluated. The agents with dominance index $I_d=0$ form a set $X_k$ of non-dominated solutions. Hence, problem (1) translates into finding a series of sets $X_k$ such that $X_k \rightarrow X_g$ for $k \rightarrow k_{\max}$ with $k_{\max}$ possibly a finite number.

The position of each agent in $D$ is updated through a number of heuristics. Some are called *collaborative actions*, because they are derived from the interaction of at least two agents and involve the entire population at one time. The general collaborative heuristic can be expressed in the following form

$$\mathbf{x}_k = \mathbf{x}_k + S(\mathbf{x}_k + \mathbf{u}_k)\mathbf{u}_k \tag{5}$$

where $\mathbf{u}_k$ depends on the other agents in the population and $S$ is a selection function which yields 0 if the candidate point $\mathbf{x}_k+\mathbf{u}_k$ is not selected or 1 if it is selected (section 3.2). In this implementation, a candidate point is selected if its dominance index is better or equal than the one of $\mathbf{x}_k$. A first restart mechanism is then implemented to avoid crowding. This restart mechanism is taken from reference [**9**] and prevents the agents from overlapping or getting too close. It is governed by the crowding factor $w_c$ that defines the minimum acceptable normalized distance between agents. Note that, this heuristic increases the uniform sampling rate of $D$, when activated, thus favouring exploration. On the other hand, by setting $w_c$ small, the agents are more directed towards local convergence.

After all the collaborative and restart actions have been implemented, the resulting updated population $P_k$ is ranked according to $I_d$ and split in two subpopulations: $P_k^u$ and $P_k^l$. The agents in each subpopulation implement sets of, so called, *individualistic actions* to collect samples of the surrounding space and to modify their current location. In particular, the last $n_{\text{pop}} - f_e n_{\text{pop}}$ agents belong to $P_k^u$ and implement heuristics that can be expressed in a form similar to equation (5) but with $\mathbf{u}_k$ that depends only on $\mathbf{x}_k$.

The remaining $f_e n_{\text{pop}}$ agents belong to $P_k^l$ and implement a mix of actions that aim at either improving their location or exploring the neighbourhood $N_\rho(\mathbf{x}_{i,k})$, with $i=1,\ldots, f_e n_{\text{pop}}$. $N_\rho(\mathbf{x}_{i,k})$ is a hyperectangle centred in $\mathbf{x}_{i,k}$. The intersection $N_\rho(\mathbf{x}_{i,k}) \cap D$ represents the local region around agent $\mathbf{x}_{i,k}$ that one wants to explore. The *size* of $N_\rho(\mathbf{x}_{i,k})$ is specified by the value $\rho(\mathbf{x}_{i,k})$: the $i$th edge of $N_\rho$ has length $2\rho(\mathbf{x}_{i,k}) \max\{b_j^u - x_{i,k}[j], x_{i,k}[j] - b_j^l\}$.

The agents in $P_k^l$ generate a number of perturbed solutions $\mathbf{y}_s$ for each $\mathbf{x}_{i,k}$, with $s=1,\ldots, s_{\max}$. These solutions are collected in a local archive $A_l$ and a dominance index is computed for all the elements in $A_l$. If at least one element $\mathbf{y}_s \in A_l$ has $I_d=0$, then $\mathbf{x}_{i,k} \leftarrow \mathbf{y}_s$. If multiple elements of $A_l$ have $I_d=0$, then the one with the largest variation, with respect to $\mathbf{x}_{i,k}$, in criteria space is taken. Figure 1(a) shows three agents (circles) with a set of locally generated samples (stars) in their respective neighbourhoods (dashed square). The arrows indicate the direction of motion of each agent. The figure show also the local archive for the first agent $A_{l_1}$ and the target global archive $X_g$.

The $\rho$ value associated to an agent is updated at each iteration according to the rule devised in reference [**9**]. Furthermore, a value $s(\mathbf{x}_{i,k})$ is associated to each agent $\mathbf{x}_{i,k}$ to specify the number of samples allocated to the exploration of $N_\rho(\mathbf{x}_{i,k})$. This value is updated at each iteration according to the rule devised in reference [**9**].

The adaptation of $\rho$ is introduced to allow the agents to self-adjust the neighbourhood removing the need to set *a priori* the appropriate size of $N_\rho(\mathbf{x}_{i,k})$. The consequence of this adaptation is an intensification of the local search by some agents, while the others are still exploring. In this respect, MACS works opposite to variable neighbourhood search heuristics, where the neighbourhood is adapted to improve global exploration, and differently than basin hopping heuristics in which the neighbourhood is fixed. Similarly, the adaptation of $s(\mathbf{x}_{i,k})$ avoids setting *a priori* an arbitrary number of

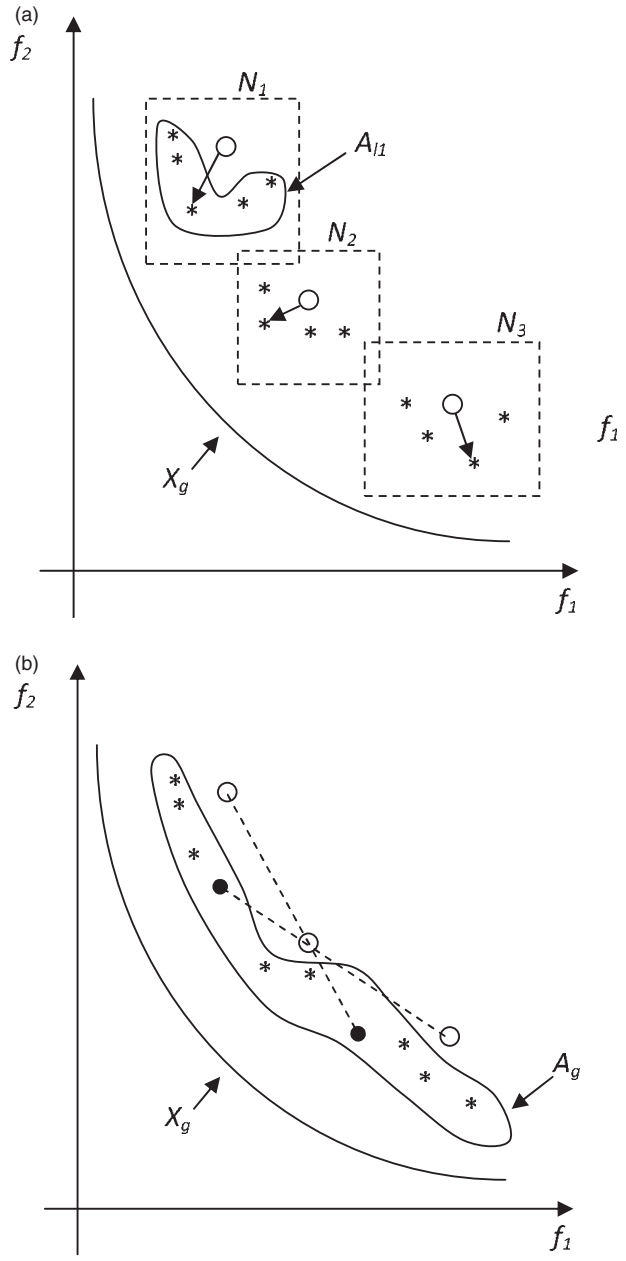


**Fig. 1** Illustration of the (a) local moves and archive and (b) global moves and archive

individualistic moves and has the effect of avoiding an excessive sampling of $N_\rho(\mathbf{x}_{i,k})$ when ρ is small. The value of $s(\mathbf{x}_{i,k})$ is initialized to the maximum number of allowable individualistic moves $s_{\max}$. The value of $s_{\max}$ is here set equal to the number of dimensions $n$. This choice is motivated by the fact that a gradient-based method would evaluate the function a minimum of $n$ times to compute an approximation of the gradient with finite differences. Note that the set of individualistic actions allows the agents to independently move towards and within the set $X$, although no specific mechanism is defined as in reference [**10**]. On the other hand, the mechanisms proposed in reference [**10**] could further improve the local search and will be the subject of a future investigation.

All the elements of $X_k$ found during one generation are stored in a global archive $A_g$. The elements in $A_g$ are a selection of the elements collected in all the local archives. Figure 1(b) illustrates three agents performing two social actions that yield two samples (black dots). The two samples together with the non-dominated solutions coming from the local archive form the global archive. The archive $A_g$ is used to implement an attraction mechanism that improves the convergence of the worst agents (section 3.4.1). During the global archiving process, a second restart mechanism that reinitializes a portion of the population (*bubble restart*) is implemented. Even this second restart mechanism is taken from reference [**9**] and avoids that, if $\rho$ collapses to zero, the agent keeps on sampling a null neighbourhood.

Note that, within the MACS framework, other strategies can be assigned to the agents to evaluate their moves in the case of multiple objective functions, for example a decomposition technique [**11**]. However, in this article, we develop the algorithm based only on the use of the dominance index.

### 3.2 Collaborative actions

Collaborative actions define operations through which information is exchanged between pairs of agents. Consider a pair of agents $\mathbf{x}_1$ and $\mathbf{x}_2$, with $\mathbf{x}_1 \prec \mathbf{x}_2$. One of the two agents is selected at random in the worst half of the current population (from the point of view of the property $I_d$), while the other is selected at random from the whole population. Then, three different actions are performed. Two of them are defined by adding to $\mathbf{x}_1$ a step $\mathbf{u}_k$ defined as follows

$$\mathbf{u}_k = \alpha r^t (\mathbf{x}_2 - \mathbf{x}_1) \tag{6}$$

and corresponding to: *extrapolation* on the side of $\mathbf{x}_1$ ($\alpha = -1$, $t = 1$), with the further constraint that the result must belong to the domain $D$ (i.e. if the step $\mathbf{u}_k$ leads out of $D$, its size is reduced until we get back to $D$); *interpolation* ($\alpha = 1$), where a random point between $\mathbf{x}_1$ and $\mathbf{x}_2$ is sampled. In the latter case, the shape parameter $t$ is defined as follows

$$t = 0.75 \frac{s(\mathbf{x}_1) - s(\mathbf{x}_2)}{s_{\max}} + 1.25 \tag{7}$$

The rationale behind this definition is that we are favouring moves, which are closer to the agent with a higher fitness value if the two agents have the same $s$

**Algorithm 1** Main MACS algorithm

Initialize a population $P_0$ of $n_{\text{pop}}$ agents in $D$, $k=0$, number of function evaluations $n_{\text{eval}}=0$, maximum number of function evaluations $N_e$, crowding factor $w_c$
**for all** $i=1,\ldots, n_{\text{pop}}$ **do**
  $\mathbf{x}_{i,k}=\mathbf{x}_{i,k}+S(\mathbf{x}_{i,k}+\mathbf{u}_{i,k})\mathbf{u}_{i,k}$
**end for**
Rank solutions in $P_k$ according to $I_d$
Re-initialize crowded agents according to the single agent restart mechanism
**for all** $i=f_e n_{\text{pop}},\ldots, n_{\text{pop}}$ **do**
  Generate $n_p$ mutated copies of $\mathbf{x}_{i,k}$.
  Evaluate the dominance of each mutated copy $\mathbf{y}_p$ against $\mathbf{x}_{i,k}$, with $p=1,\ldots, n_p$
  **if** $\exists p | \mathbf{y}_p \prec \mathbf{x}_{i,k}$ **then**
    $\bar{p} = \arg\max_p \|\mathbf{y}_p - \mathbf{x}_{i,k}\|$
    $\mathbf{x}_{i,k} \leftarrow \mathbf{y}_{\bar{p}}$
  **end if**
**end for**
**for all** $i=1,\ldots, f_e n_{\text{pop}}$ **do**
  Generate $s < s_{\max}$ individual actions $\mathbf{u}_s$ such that $\mathbf{y}_s=\mathbf{x}_{i,k}+\mathbf{u}_s$
  **if** $\exists s | \mathbf{y}_s \prec \mathbf{x}_{i,k}$ **then**
    $\bar{s} = \arg\max_s \|\mathbf{y}_s - \mathbf{x}_{i,k}\|$
    $\mathbf{x}_{i,k} \leftarrow \mathbf{y}_{\bar{s}}$
  **end if**
  Store candidate elements $\mathbf{y}_s$ in the local archive $A_l$
  Update $\rho(\mathbf{x}_{i,k})$ and $s(\mathbf{x}_{i,k})$
**end for**
Form $P_k = P_k^l \bigcup P_k^u$ and $\hat{A}_g = A_g \bigcup A_l \bigcup P_k$
Compute $I_d$ of all the elements in $\hat{A}_g$
$A_g = \{\mathbf{x} | \mathbf{x} \in \hat{A}_g \wedge I_d(\mathbf{x}) = 0 \wedge \|\mathbf{x} - \mathbf{x}_{A_g}\| > w_c\}$
Re-initialize crowded agents in $P_k$ according to the second restart mechanism
Compute attraction component to $A_g$ for all $\mathbf{x}_{i,k} \in P_k \setminus X_k$
$k=k+1$
**Termination** Unless $n_{\text{eval}} > N_e$, GoTo Step 2

value, while in the case where the agent with the highest fitness value has a $s$ value much lower than that of the other agent, we try to move away from it, because a small $s$ value indicates that improvements close to the agent are difficult to detect.

The third operation is the *recombination* operator, a *single-point crossover*, where given the two agents: we randomly select a component $j$; split the two agents into two parts, one from component 1 to component $j$ and the other from component $j+1$ to component $n$; and then, we combine the two parts of each of the agents in order to generate two new solutions. The three operations give rise to four new samples, denoted by $\mathbf{y}_1$, $\mathbf{y}_2$, $\mathbf{y}_3$, and $\mathbf{y}_4$. Then, $I_d$ is computed for the set $\mathbf{x}_2$, $\mathbf{y}_1$, $\mathbf{y}_2$, $\mathbf{y}_3$, $\mathbf{y}_4$. The element with $I_d=0$ becomes the new location of $\mathbf{x}_2$ in $D$.

### 3.3 Individualistic actions

Once the collaborative actions have been implemented, each agent in $P_k^u$ is mutated a number of times: the lower the ranking the higher the number of mutations. The mutation mechanisms is not different from the single objective case, but the selection is modified to use the dominance index rather than the objective values.

A mutation is simply a random vector $\mathbf{u}_k$ such that $\mathbf{x}_{i,k}+\mathbf{u}_k \in D$. All the mutated solution vectors are then compared to $\mathbf{x}_{i,k}$, i.e. $I_d$ is computed for the set made of the mutated solutions and $\mathbf{x}_{i,k}$. If at least one element $\mathbf{y}_p$ of the set has $I_d=0$, then $\mathbf{x}_{i,k} \leftarrow \mathbf{y}_p$. If multiple elements of the set have $I_d=0$, then the one with the largest variation, with respect to $\mathbf{x}_{i,k}$, in criteria space is taken.

Each agent in $P_k^l$ performs at most $s_{\max}$ of the following individualistic actions: *inertia*, *differential*, *random with line search*. The overall procedure is summarized in Algorithm 2 and each action is described in detail in the following subsections.

#### 3.3.1 Inertia

If agent $i$ has improved from generation $k-1$ to generation $k$, then it follows the direction of the improvement (possibly until it reaches the border of $D$), i.e. it performs the following step

$$\mathbf{y}_s = \mathbf{x}_{i,k} + \bar{\lambda}\Delta_I \tag{8}$$

where $\bar{\lambda} = \min\{1, \max\{\lambda : \mathbf{y}_s \in D\}\}$ and $\Delta_I = (\mathbf{x}_{i,k} - \mathbf{x}_{i,k-1})$.

#### 3.3.2 Differential

This step is inspired by differential evolution [**12**]. It is defined as follows: let $\mathbf{x}_{i_1,k}$, $\mathbf{x}_{i_2,k}$, $\mathbf{x}_{i_3,k}$ be three randomly selected agents; then

$$\mathbf{y}_s = \mathbf{x}_{i,k} + \boldsymbol{e}\left[\mathbf{x}_{i_1,k} + F(\mathbf{x}_{i_3,k} - \mathbf{x}_{i_2,k})\right] \tag{9}$$

with **e**, a vector containing a random number of 0 and 1 (the product has to be intended componentwise) with probability 0.8 and $F=0.8$ in this implementation. For every component $y_s[j]$ of $\mathbf{y}_s$ that is outside the boundaries defining $D$, then $y_s[j] = r(b_j^u - b_j^l) + b_j^l$, with $r \in U(0, 1)$. Note that, although this action involves more than one agent, its outcome is only compared to the other outcomes coming from the actions performed by agent $\mathbf{x}_{i,k}$ and therefore it is considered individualistic.

### 3.3.3 Random with line search

This move realizes a local exploration of the neighbourhood $N(\mathbf{x}_{i,k})$. It generates a first random sample $\mathbf{y}_s \in N(\mathbf{x}_{i,k})$. Then, if $\mathbf{y}_s$ is not an improvement, it generates a second sample $\mathbf{y}_{s+1}$ by extrapolating on the side of the better one between $\mathbf{y}_s$ and $\mathbf{x}_{i,k}$

$$\mathbf{y}_{s+1} = \mathbf{x}_{i,k} + \bar{\lambda}\left[\alpha_2 r^t(\mathbf{y}_s - \mathbf{x}_{i,k}) + \alpha_1(\mathbf{y}_s - \mathbf{x}_{i,k})\right] \quad (10)$$

with $\bar{\lambda} = \min\{1, \max\{\lambda \ : \ \mathbf{y}_{s+1} \in D\}\}$ and where $\alpha_1$, $\alpha_2 \in \{-1, 0, 1\}$, $r \in U(0, 1)$ and $t$ is a shaping parameter which controls the magnitude of the displacement. Here, we use the parameter values $\alpha_1 = 0$, $\alpha_2 = -1$, $t=1$, which correspond to *extrapolation* on the side of $\mathbf{x}_{i,k}$, and $\alpha_1 = \alpha_2 = 1$, $t=1$, which correspond to *extrapolation* on the side of $\mathbf{y}_s$.

The outcome of the extrapolation is used to construct a second-order one-dimensional model of $I_d$. The second-order model is given by the quadratic function $f_l(\sigma) = a_1\sigma^2 + a_2\sigma + a_3$, where $\sigma$ is a coordinate along the $\mathbf{y}_{s+1} - \mathbf{y}_s$ direction. The coefficients $a_1$, $a_2$, and $a_3$ are computed so that $f_l$ interpolates the values $I_d(\mathbf{y}_s)$, $I_d(\mathbf{y}_{s+1})$, and $I_d(\mathbf{x}_{i,k})$. In particular, for $\sigma = 0$ $f_l = I_d(\mathbf{y}_s)$, $\sigma = 1$ $f_l = I_d(\mathbf{y}_{s+1})$, and for $\sigma = (\mathbf{x}_{i,k} - \mathbf{y}_s)/\|\mathbf{x}_{i,k} - \mathbf{y}_s\|$ $f_l = I_d(\mathbf{x}_{i,k})$. Then, a new sample $\mathbf{y}_{s+2}$ is taken at the minimum of the second-order model along the $\mathbf{y}_{s+1} - \mathbf{y}_s$ direction.

The position of $\mathbf{x}_{i,k}$ in $D$ is then updated with the $\mathbf{y}_s$ that has $I_d = 0$ and the longest vector difference in the criteria space with respect to $\mathbf{x}_{i,k}$. The displaced vectors $\mathbf{y}_s$ generated by the agents in $P_k^l$ are not discarded but contribute to a local archive $A_l$, one for each agent, except for the one selected to update the location of $\mathbf{x}_{i,k}$. In order to rank the $\mathbf{y}_s$, the following modified dominance index is used

$$\hat{I}_d(\mathbf{x}_{i,k}) = \left|\left\{j \mid f_j(\mathbf{y}_s) = f_j(\mathbf{x}_{i,k})\right\}\right|\kappa + \left|\left\{j \mid f_j(\mathbf{y}_s) > f_j(\mathbf{x}_{i,k})\right\}\right| \quad (11)$$

where $\kappa$ is equal to one if there is at least one component of $\mathbf{f}(\mathbf{x}_{i,k}) = [f_1, f_2, \ldots, f_m]^T$ which is better than the corresponding component of $\mathbf{f}(\mathbf{y}_s)$, and is equal to zero otherwise.

**Algorithm 2** Individual Actions in $P_k^l$

1: $s=1$, $stop=0$
2: **if** $\mathbf{x}_{i,k} \prec \mathbf{x}_{i,k-1}$ **then**
  $\mathbf{y}_s = \mathbf{x}_{i,k} + \bar{\lambda}(\mathbf{x}_{i,k} - \mathbf{x}_{i,k-1})$
  with $\bar{\lambda} = \min\{1, \max\{\lambda \ : \ \mathbf{y}_s \in D\}\}$.
3: **end if**
4: **if** $\mathbf{x}_{i,k} \prec \mathbf{y}_s$ **then**
  $s = s+1$
  $\mathbf{y}_s = \boldsymbol{e}[\mathbf{x}_{i,k} - (\mathbf{x}_{i_1,k} + (\mathbf{x}_{i_3,k} - \mathbf{x}_{i_2,k}))]$
  $\forall j y_s(j) \notin D$, $y_s(j) = r(b_u(j) - b_l(j)) + b_l(j)$,
  with $j = 1, \ldots, n$ and $r \in U(0, 1)$
5: **else** $stop=1$
6: **end if**
7: **if** $\mathbf{x}_{i,k} \prec \mathbf{y}_s$ **then**
  $s = s+1$
  Generate $\mathbf{y}_s \in N_\rho(\mathbf{x}_{i,k})$.
  Compute $\mathbf{y}_{s+1} = \mathbf{x}_{i,k} + \bar{\lambda} r^t(\mathbf{x}_{i,k} - \mathbf{y}_s)$
  with $\bar{\lambda} = \min\{1, \max\{\lambda \ : \ \mathbf{y}_s \in D\}\}$
  and $r \in U(0, 1)$.
  Compute $\mathbf{y}_{s+2} = \bar{\lambda}\sigma_{min}(\mathbf{y}_{s+1} - \mathbf{y}_s)/\|(\mathbf{y}_{s+1} - \mathbf{y}_s)\|$,
  with $\sigma_{\min} = \arg\min_\sigma \{a_1\sigma^2 + a_2\sigma + I_d(\mathbf{y}_s)\}$,
  and $\bar{\lambda} = \min\{1, \max\{\lambda \ : \ \mathbf{y}_{s+2} \in D\}\}$.
  $s = s+2$
8: **else** $stop=1$
9: **end if**
10: **Termination** Unless $s > s_{\max}$ or $stop=1$ , GoTo Step 4

Now, if for the $s$th outcome, the dominance index in equation (11) is not zero, but is lower than the number of components of the objective vector, then the agent $\mathbf{x}_{i,k}$ is only partially dominating the $s$th outcome. Among all the partially dominated outcomes with the same dominance index, we select the one that satisfies the condition

$$\min_s \left\langle \left(\mathbf{f}(\mathbf{x}_{i,k})\right) - \mathbf{f}(\mathbf{y}_s)\right), \mathbf{e}\right\rangle \quad (12)$$

where **e** is the unit vector of dimension $m$, $\mathbf{e} = \frac{[1, 1, 1, \ldots, 1]^T}{\sqrt{m}}$. All the non-dominated and selected partially dominated solutions form the local archive $A_l$.

## 3.4 The local and global archives $A_L$ and $A_G$

Since the outcomes of one agent could dominate other agents or the outcomes of other agents, at the end of each generation, every $A_l$ and the whole population $P_k$ are added to the current global archive $A_g$. The global $A_g$ contains $X_k$, the current best estimate of $X_g$. The dominance index in equation (3) is then computed for all the elements in $\hat{A}_g = A_g \bigcup_l A_l \bigcup P_k$ and only the non-dominated ones with crowding distance $\|\mathbf{x}_{i,k} - \mathbf{x}_{A_g}\| > w_c$ are preserved (where $\mathbf{x}_{A_g}$ is an element of $A_g$).

### 3.4.1 Attraction

The archive $A_g$ is used to direct the movements of those agents that are outside $X_k$. All agents, for which $I_d \neq 0$ at step $k$, are assigned the position of

the elements in $A_g$ and their inertia component is recomputed as

$$\Delta_I = r(\mathbf{x}_{A_g} - \mathbf{x}_{i,k}) \tag{13}$$

More precisely, the elements in the archive are ranked according to their reciprocal distance or crowding factor. Then, every agent for which $I_d \neq 0$ picks the least crowded element $\mathbf{x}_{A_g}$ not already picked by any other agent.

### 3.5 Stopping rule

The search is stopped when a prefixed number $N_e$ of function evaluations is reached. At termination of the algorithm, the whole final population is inserted into the archive $A_g$.

## 4 COMPARATIVE TESTS

The proposed optimization approach was implemented in a software code, in MATLAB, called MACS. In previous works [**8**, **9**], MACS was tested on single objective optimization problems related to space trajectory design, showing good performances. In this study, MACS was tested at first on a number of standard problems, found in literature, and then on three typical space trajectory optimization problems.

This article extends the results presented in Vasile and Zuiani [**13**] by adding a broader suite of algorithms for multi-objective optimization to the comparison and a different formulation of the performance metrics.

### 4.1 Tested algorithms

MACS was compared against a number of state-of-the-art algorithms for multi-objective optimization. For this analysis, it was decided to take the basic version of the algorithms that is available online. Further developments of the basic algorithms have not been considered in this comparison and will be included in future works. Note that, all the algorithms selected for this comparative analysis use Pareto dominance as selection criterion.

The tested algorithms are: NSGA-II [**14**], multiple objective particle swarm optimization (MOPSO) [**15**], Pareto archived evolution strategy (PAES) [**16**], and multiple trajectory search (MTS) [**17**]. A short description of each algorithm with their basic parameters follows.

#### *4.1.1 NSGA-II*

Non-dominated sorting genetic algorithm (NSGA-II) is a genetic algorithm which uses the concept of dominance class (or depth) to rank the population. A crowding factor is then used to rank the individuals within each dominance class. Optimization starts from a randomly generated initial population. The individuals in the population are sorted according to their level of Pareto dominance with respect to other individuals. To be more precise, a fitness value equal to 1 is assigned to the non-dominated individuals. Non-dominated individuals form the first layer (or class). Those individuals dominated only by members of the first layer form the second one and are assigned a fitness value of 2, and so on. In general, for dominated individuals, the fitness is given by the number of dominating layers plus 1. A crowding factor is then assigned to each individual in a given class. The crowding factor is computed as the sum of the Euclidean distances, in criteria space, with respect to other individuals in the same class, divided by the interval spanned by the population along each dimension of the objective space. Inside each class, the individuals with the higher value of the crowding parameter obtain a better rank than those with a lower one.

At every generation, binary tournament selection, recombination, and mutation operators are used to create an offspring of the current population. The combination of the two is then sorted according to dominance first and then to crowding. The non-dominated individuals with the lowest crowding factor are then used to update the population.

The parameters to be set are the size of the population, the number of generations, the cross-over and mutation probability, $p_c$ and $p_m$, and distribution indexes for cross-over and mutation, $\eta_c$ and $\eta_m$, respectively. Three different ratios between population size and number of generations were considered: 0.08, 0.33, and 0.75. The values $p_c$ and $p_m$ were set to 0.9 and 0.2, respectively and kept constant for all the tests. The values, 5, 10, and 20, were considered for $\eta_c$, while tests were run for values of $\eta_m$ equal to 5, 25, and 50.

#### *4.1.2 Multiple objective particle swarm optimization*

Multiple objective particle swarm optimization is an extension of particle swarm optimization (PSO) to multi-objective problems. Pareto dominance is introduced in the selection criteria for the candidate solutions to update the population. MOPSO features an external archive which stores all the non-dominated solutions and at the same time is used to guide the search process of the swarm. This is done by introducing the possibility to direct the movement of a particle towards one of the less-crowded solutions in the archive. The solution space is subdivided into hypercubes through an adaptive grid. The solutions

in the external archive are thus reorganized in these hypercubes. The algorithm keeps track of the crowding level of each hypercube and promotes movements towards less-crowded areas. In a similar manner, it also gives priority to the insertion of new non-dominated individuals in less-crowded areas if the external archive has already reached its predefined maximum size. For MOPSO, three different ratios between population size and number of generations were tested: 0.08, 0.33, and 0.75. It was also tested with three different numbers of subdivisions of the solution space: 10, 30, and 50. The inertia component in the motion of the particles was set to 0.4 and the weights of the social and individualistic components were se to 1.

#### *4.1.3 Pareto archived evolution strategy*

Pareto archived evolution strategy is a $(1+1)$ evolution strategy with the addition of an external archive to store the current best approximation of the Pareto front. It adopts a population of only a single chromosome which, at every iteration, generates a mutated copy. The algorithm then preserves the non-dominated one between the parent and the candidate solution. If none of the two dominates the other, the algorithm then checks their dominance index with respect to the solutions in the archive. If also this comparison is inconclusive, then the algorithm selects the one which resides in the less-crowded region of the objective space. To keep track of the crowding level of the objective space, the latter is subdivided in an $n$-dimensional grid. Every time a new solution is added to (or removed from) the archive, the crowding level of the corresponding grid cell is updated. PAES has two main parameters that need to be set, the number of subdivisions in the space grid and the mutation probability. Values of 1, 2, and 4 were used for the former and 0.6, 0.8, and 0.9 for the latter.

#### *4.1.4 Multiple trajectory search*

Multiple trajectory search is an algorithm based on pattern search. The algorithm first generates a population of uniformly distributed individuals. At every iteration, a local search is performed by a subset of individuals. Three different search patterns are included in the local search: the first is a search along the direction of each decision variable with a fixed step length; the second is analogous but the search is limited to one-fourth of all the possible search directions; the third one also searches along each direction but selects only solutions, which are evenly spaced on each dimension within a predetermined upper bound and lower bound. At each iteration and for each individual, the three search patterns are tested with few function evaluations to select the one which generates the best candidate solutions for the current individual. The selected one is then used to perform the local search which will update the individual itself. The step length along each direction, which defines the size of the search neighbourhood for each individual, is increased if the local search generated a non-dominated child and is decreased otherwise. When the neighbourhood size reaches a predetermined minimum value, it is reset to 40 per cent of the size of the global search space. The non-dominated candidate solutions are then used to update the best approximation of the global Pareto front. MTS was tested with a population size of 20, 40, and 80 individuals.

### 4.2. Performance metrics

Two metrics were defined to evaluate the performance of the tested multi-objective optimizers

$$M_{\text{spr}} = \frac{1}{M_p}\sum_{i=1}^{M_p}\min_{j\in N_p} 100\left\|\frac{\mathbf{f}_j-\mathbf{g}_i}{\mathbf{g}_i}\right\| \tag{14}$$

$$M_{\text{conv}} = \frac{1}{N_p}\sum_{i=1}^{N_p}\min_{j\in M_p} 100\left\|\frac{\mathbf{g}_j-\mathbf{f}_i}{\mathbf{g}_j}\right\| \tag{15}$$

where $M_p$ is the number of elements, with objective vector $\mathbf{g}$, in the true global Pareto front and $N_p$ the number of elements, with objective vector $\mathbf{f}$, in the Pareto front that a given algorithm is producing. Although similar, the two metrics are measuring two different things: $M_{\text{spr}}$ is the sum, over all the elements in the global Pareto-front, of the minimum distance of all the elements in the Pareto front $N_p$ from the $i^{th}$ element in the global Pareto front. $M_{\text{conv}}$, instead, is the sum, over all the elements in the Pareto front $N_p$, of the minimum distance of the elements in the global Pareto front from the $i$th element in the Pareto front $N_p$.

Therefore, if $N_p$ is only a partial representation of the global Pareto front but is a very accurate partial representation, then metric $M_{\text{spr}}$ would give a high value and metric $M_{\text{conv}}$ a low value. If both metrics are high, then the Pareto front $N_p$ is partial and poorly accurate. The index $M_{\text{conv}}$ is similar to the mean Euclidean distance [**15**], although in $M_{\text{conv}}$, the Euclidean distance is normalized with respect to the values of the objective functions, while $M_{\text{spr}}$ is similar to the generational distance [**18**], although even for $M_{\text{spr}}$ the distances are normalized.

Given $n$ repeated runs of a given algorithm, we can define two performance indexes: $p_{\text{conv}} = P(M_{\text{conv}} < tol_{\text{conv}})$ or the probability that the index $M_{\text{conv}}$ achieves a value less than the threshold $tol_{\text{conv}}$ and $p_{\text{spr}} = P(M_{\text{spr}} < tol_{\text{spr}})$ or the probability that the

index $M_{spr}$ achieves a value less than the threshold $tol_{conv}$.

According to the theory developed in references [**7**, **19**], 200 runs are sufficient to have a 95 per cent confidence that the true values of $p_{conv}$ and $p_{spr}$ are within a ±5 per cent interval containing their estimated value.

Performance index (14) and (15) tend to uniformly weigh every part of the front. This is not a problem for index (14) but if only a relatively small portion of the front is missed the value of performance index (15) might be only marginally affected. For this reason, we slightly modified the computation of the indexes by taking only the $M_P^*$ and $N_P^*$ solutions with a normalized distance in criteria space that was higher than $10^{-3}$.

The global fronts used in the three space tests were built by taking a selection of about 2000 equispaced, in criteria space, non-dominated solutions coming from all the 200 runs of all the algorithms.

### 4.3. Preliminary Test Cases

For the preliminary tests, two sets of functions, taken from the literature [**14**, **15**], were used and the performance of MACS was compared to the results in [**14**, **15**]. Therefore, for this set of tests, MTS was not included in the comparison. The function used in this section can be found in Table 1.

The first three functions were taken from reference [**15**]. Test cases *Deb* and *Scha* are two examples of disconnected Pareto fronts, *Deb*2 is an example of problem presenting multiple local Pareto fronts, 60 in this two-dimensional case.

The last three functions are instead taken from reference [**14**]. Test case *ZDT*2 has a concave Pareto front with a moderately high-dimensional search space. Test case *ZDT*6 has a concave Pareto front but because of the irregular nature of $f_1$ there is a strong bias in the distribution of the solutions. Test case, *ZDT4*, with dimension 10, is commonly

**Table 1** Multi-objective test functions

| | |
|---|---|
| *Scha* | $f_2 = (x-5)^2$ |
| $x \in [-5, 10]$ | $f_1 = \begin{cases} -x & if \quad x \le 1 \\ -2+x & if \quad 1 < x < 3 \\ 4-x & if \quad 3 < x \le 4 \\ -4+x & if \quad x > 4 \end{cases}$ |
| *Deb* | $f_1 = x_1$ |
| $x_1, x_2 \in [0, 1]$ | $f_2 = (1+10x_2)\left[1 - \left(\frac{x_1}{1+10x_2}\right)^{\alpha} - \frac{x_1}{1+10x_2}\sin(2\pi q x_1)\right]$ |
| $\alpha = 2$; $q = 4$ | |
| *Deb2* | $f_1 = x_1$ |
| $x_1 \in [0, 1]$ | $f_2 = g(x_1, x_2)h(x_1, x_2)$; $g(x_1, x_2) = 11 + x_2^2 - 10\cos(2\pi x_2)$ |
| $x_2 \in [-30, 30]$ | $h(x_1, x_2) = \begin{cases} 1 - \sqrt{\frac{f_1}{g}} & if f_1 \le g \\ 0 & otherwise \end{cases}$ |
| *ZDT2* | $g = 1 + \frac{9}{n-1}\sum_{i=2}^{n} x_i$ |
| $x_i \in [0, 1]$; | $h = 1 - (\frac{f_1}{g})^2$ |
| $i = 1, \ldots, n$ | $f_1 = x_1$; $f_2 = gh$ |
| $n = 30$ | |
| *ZDT4* | $g = 1 + 10(n-1) + \sum_{i=2}^{n}[x_i^2 - 10\cos(2\pi q x_i)]$; |
| $x_1 \in [0, 1]$; | $h = 1 - \sqrt{\frac{f_1}{g}}$ |
| $x_i \in [-5, 5]$; | $f_1 = x_1$; $f_2 = gh$ |
| $i = 2, \ldots, n$ | |
| $n = 10$ | |
| *ZDT6* | $g = 1 + 9\sqrt[4]{\frac{\sum_{i=2}^{n} x_i}{n-1}}$ |
| $x_i \in [0, 1]$; | $h = 1 - (\frac{f_1}{g})^2$ |
| $i = 1, \ldots, n$ | $f_1 = 1 - \exp(-4x_1)\sin^6(6\pi x_1)$; $f_2 = gh$ |
| $n = 10$ | |

recognized as one of the most challenging problems since it has $21^9$ different local Pareto fronts of which only one corresponds to the global Pareto-optimal front.

As a preliminary proof of the effectiveness of MACS, the average Euclidean distance of 500 uniformly spaced points on the true optimal Pareto front from the solutions stored in $A_g$ by MACS was computed and compared to known results in the literature. MACS was run 20 times to have a sample comparable to the one used for the other algorithms. The global archive was limited to 200 elements to be consistent with reference [**14**]. The value of the crowding factor $w_c$, the threshold $\rho_{\text{tol}}$ and the convergence $\rho_{\min}$ were kept constant to 1e-5 in all the cases to provide good local convergence.

To be consistent with reference [**15**], on *Deb*, *Scha* and, *Deb*2, MACS was run, respectively, for 4000, 1200, and 3200 function evaluations. Only two agents were used for these lower dimensional cases, with $f_e = 1/2$. On test cases *ZDT*2, *ZDT*4, and *ZDT*6, MACS was run for a maximum of 25 000 function evaluations to be consistent with reference [**14**], with three agents and $f_e = 2/3$ for *ZDT*2 and four agents and $f_e = 3/4$ on *ZDT*4 and *ZDT*6.

The results on *Deb*, *Scha*, and *Deb*2 can be found in Table 2, while the results on *ZDT*2, *ZDT*4, and *ZDT*6 can be found in Table 3.

On all the smaller dimensional cases, MACS performs comparably to MOPSO and better than PAES. It also performs than NSGA-II on *Deb* and *Deb2*. On *Scha*, MACS performs apparently worse than NSGA-II, although after inspection one can observe that all the elements of the global archive $A_g$ belong to the Pareto front but not uniformly distributed, hence the higher value of the Euclidean distance. On the higher dimensional cases, MACS performs comparably to NSGA-II on ZDT2 but better than all the others on ZDT4 and ZDT6. Note, in particular, the improved performance on ZDT4.

On the same six functions, a different test was run to evaluate the performance of different variants of MACS. For all variants, the number of agents, $f_e$, $w_c$, $\rho_{\text{tol}}$, and $\rho_{\min}$ was set as before, but instead of the mean Euclidean distance, the success rates $p_{\text{conv}}$ and $p_{\text{spr}}$ were measured for each variant. The number of function evaluations for *ZDT*2, *ZDT*4, *ZDT*6, and *Deb*2 is the same as before, while for *Scha* and *Deb* it was reduced, respectively, to 600 and 1000 function evaluations given the good performance of MACS already for this number of function evaluations. Each run was repeated 200 times to have good confidence in the values of $p_{\text{conv}}$ and $p_{\text{spr}}$.

Four variants were tested and compared to the full version of MACS. Variant MACS no local does not implement the individualistic moves and the local archive, variant MACS $\rho = 1$ has no adaptivity on the neighbourhood, its size is kept fixed to 1,variant MACS $\rho = 0.1$ has the size of the neighbourhood fixed to 0.1, variant MACS no attraction has the attraction mechanisms not active.

**Table 2** Comparison of the average Euclidean distances between 500 uniformly space points on the optimal Pareto front for various optimization algorithms: smaller dimension test problems

| Approach | Deb2 | Scha | Deb |
|---|---|---|---|
| MACS | 1.542e−3 (5.19e−4) | 3.257e−3 (5.61e−4) | 7.379e−4 (6.36e−5) |
| NSGA-II | 0.094 644 (0.117 608) | 0.001 594 (0.000 122) | 0.002 536 (0.000 138) |
| PAES | 0.259 664 (0.573 286) | 0.070 003 (0.158 081) | 0.002 881 (0.002 13) |
| MOPSO | 0.001 161 1 (0.000 720 5) | 0.001 473 96 (0.000 201 78) | 0.002 057 (0.000 286) |

**Table 3** Comparison of the average Euclidean distances between 500 uniformly space points on the optimal Pareto front for various optimization algorithms: larger dimension test problems

| Approach | ZDT2 | ZDT4 | ZDT6 |
|---|---|---|---|
| MACS | 9.0896e−4 (4.0862e−5) | 0.0061 (0.0133) | 0.0026 (0.0053) |
| NSGA-II | 0.000 824 (<1e−5) | 0.513 053 (0.118 460) | 0.296 564 (0.013 135) |
| PAES | 0.126 276 (0.036 877) | 0.854 816 (0.527 238) | 0.085 469 (0.006 644) |

The result can be found in Table 4. The values of $tol_{\text{conv}}$ and $tol_{\text{spr}}$ are, respectively, 0.001 and 0.0035 for *ZDT*2, 0.003 and 0.005 for *ZDT*4, 0.001 and 0.025 for *ZDT*6, 0.0012 and 0.035 for *Deb*, 0.0013 and 0.04 for *Scha*, 0.0015 and 0.0045 for *Deb*2. These thresholds were selected to highlight the differences among the various variants.

The table shows that the adaptation mechanism is beneficial in some cases although, in others, fixing the value of $\rho$ might be a better choice. This depends on the problem and a general rule is difficult to derive at present. Other adaptation mechanisms could further improve the performance.

The use of individualistic actions coupled with a local archive is instead fundamental, so is the use of the attraction mechanism. Note, however, how the attraction mechanism penalizes the spreading on biassed problems like *ZDT*6, this is expected as it accelerates convergence.

**Table 4** Comparison of different versions of MACS

| Approach | Metric | ZDT2(%) | ZDT4(%) | ZDT6(%) | Scha(%) | Deb(%) | Deb2(%) |
|---|---|---|---|---|---|---|---|
| MACS | $p_{\text{conv}}$ | 83.5 | 75 | 77 | 73 | 70.5 | 60 |
| | $p_{\text{spr}}$ | 22.5 | 28 | 58.5 | 38.5 | 83 | 67.5 |
| MACS | $p_{\text{conv}}$ | 14 | 0 | 45 | 0.5 | 72.5 | 11 |
| no local | $p_{\text{spr}}$ | 1 | 0 | 34 | 0 | 4 | 15 |
| MACS | $p_{\text{conv}}$ | 84 | 22 | 78 | 37 | 92 | 21 |
| $\rho = 1$ | $p_{\text{spr}}$ | 22 | 7 | 63 | 0 | 54 | 38 |
| MACS | $p_{\text{conv}}$ | 56 | 42 | 57 | 78 | 85 | 42 |
| $\rho = 0.1$ | $p_{\text{spr}}$ | 5 | 15 | 61 | 88 | 94.5 | 74 |
| MACS | $p_{\text{conv}}$ | 21 | 0.5 | 14.5 | 0.5 | 88.5 | 0 |
| No attraction | $p_{\text{spr}}$ | 0 | 0.5 | 78.5 | 0 | 0 | 0 |

### 4.4. Application to space trajectory design

In this section, we present the application of MACS to three space trajectory problems: a two-impulse orbit transfer from a low Earth orbit (LEO) to a high-eccentricity Molniya-like orbit, a three-impulse transfer from a LEO to geostationary Earth orbit (GEO) and a multi gravity assist transfer to Saturn equivalent to the transfer trajectory of the Cassini mission. The first two cases are taken from the work of Minisci and Avanzini [**7**].

In the two-impulse case, the spacecraft departs at time $t_0$ from a circular orbit around the Earth (the gravity constant is $\mu_E = 3.986\,010^5\,\text{km}^3/\text{s}^2$) with radius $r_0 = 6721$ km and at time $t_f$ is injected into an elliptical orbit with eccentricity $e_T = 0.667$ and semi-major axis $a_T = 26\,610$ km. The transfer arc is computed as the solution of a Lambert's problem [**20**] and the objective functions are the transfer time $T = t_f - t_0$ and the sum of the two norms of the velocity variations at the beginning and at the end of the transfer arc $\Delta v_{\text{tot}}$. The objectives are functions of the solution vector $\mathbf{x} = [t_0 \;\; t_f]^T \in D \subset \mathbb{R}^2$. The search space $D$ is defined by the following intervals $t_0 \in [0\;\; 10.8]$ and $t_f \in [0.03\;\; 10.8]$.

In the three-impulse case, the spacecraft departs at time $t_0$ from a circular orbit around the Earth with radius $r_0 = 7000$ km and after a transfer time $T = t_1 + t_2$ is injected into a circular orbit with radius $r_f = 42\,000$. An intermediate manoeuvre is performed at time $t_0 + t_1$ and at position defined in polar coordinates by the radius $r_1$ and the angle $\theta_1$. The objective functions are the total transfer time $T$ and the sum of the three impulses $\Delta v_{\text{tot}}$. The solution vector in this case is $\mathbf{x} = [t_0, t_1, r_1, \theta_1, t_f]^T \in D \subset \mathbb{R}^5$. The search space $D$ is defined by the following intervals $t_0 \in [0\;\; 1.62]$, $t_1 \in [0.03 \;\; 21.54]$, $r_1 \in [7010 \;\; 105410]$, $\theta_1 \in [0.01\;\; 2\pi - 0.01]$, and $t_2 \in [0.03\;\; 21.54]$.

The Cassini case consists of five transfer arcs connecting a departure planet, the Earth, to the destination planet, Saturn, through a sequence of swing-by's with the planets: Venus, Venus, Earth, and Jupiter. Each transfer arc is computed as the solution of a Lambert's problem [**21**] given the departure time from planet $P_i$ and the arrival time at planet $P_{i+1}$. The solution of the Lambert's problems yields the required incoming and outgoing velocities at each swing-by planet $v_{\text{in}}$ and $v_{r\text{out}}$. The swing-by is modelled through a linked-conic approximation with powered manoeuvres [**22**], i.e. the mismatch between the required outgoing velocity $v_{r\text{out}}$ and the achievable outgoing velocity $v_{a\text{out}}$ is compensated through a $\Delta v$ manoeuvre at the pericentre of the gravity assist hyperbola. The whole trajectory is completely defined by the departure time $t_0$ and the transfer time for each leg $T_i$, with $i = 1, \ldots, 5$. The normalized radius of the pericentre $r_{p,i}$ of each swing-by hyperbola is derived *a posteriori* once each powered swing-by manoeuvre is computed. Thus, a constraint on each pericentre radius has to be introduced during the search for an optimal solution. In order to take into account this constraint, one of the objective functions is augmented with the weighted violation of the constraints

$$f(\boldsymbol{x}) = \Delta v_0 + \sum_{i=1}^{4} \Delta v_i + \Delta v_f + \sum_{i=1}^{4} w_i (r_{p,i} - r_{p\min,i})^2 \tag{16}$$

for a solution vector $\mathbf{x} = [t_0, T_1, T_2, T_3, T_4, T_5]^T$. The objective functions are, the total transfer time $T = \sum_i^5 T_i$ and $f(\mathbf{x})$. The minimum normalized pericentre radii are $r_{\text{pmin},1} = 1.0496$, $r_{\text{pmin},2} = 1.0496$, $r_{\text{pmin},3} = 1.0627$, and $r_{\text{pmin},4} = 9.3925$. The search space $D$ is defined by the following intervals: $t_0 \in [-1000,\; 0]$MJD2000, $T_1 \in [30,\; 400]$d, $T_2 \in [100,\; 470]$d, $T_3 \in [30,\; 400]$d, $T_4 \in [400,\; 2000]$d, and $T_5 \in [1000,\; 6000]$d. The best known solution for the single objective minimization of $f(\mathbf{x})$ is $f_{\text{best}} = 4.9307$ km/s, with $\mathbf{x}_{\text{best}} = [-789.753,\; 158.2993,\; 449.3859,\; 54.7060,\; 1024.5896,\; 4552.7054]^T$.

#### *4.4.1 Test results*

For this second set of tests, each algorithm was run for the same number of function evaluations. In particular, consistent with the tests performed in the work

of Minisci and Avanzini [7], we used 2000 function evaluations for the two-impulse case and 30 000 for the three-impulse case. For the Cassini case, instead, the algorithms were run for 180 000, 300 000, and 600 000 function evaluations.

Note that the version of all the algorithms used in this second set of tests is the one that is freely available online, written in C/C++. We tried in all cases to stick to the available instructions and recommendations by the author to avoid any bias in the comparison.

The threshold values for the two-impulse cases was taken from reference [7] and is $tol_{\text{conv}} = 0.1$, $tol_{\text{spr}} = 2.5$. For the three-impulse, case instead we considered $tol_{\text{conv}} = 5.0$, $tol_{\text{spr}} = 5.0$. For the Cassini case, we used $tol_{\text{conv}} = 0.75$, $tol_{\text{spr}} = 5$, instead. These values were selected after looking at the dispersion of the results over 200 runs. Lower values would result in a zero value of the performance indexes of all the algorithms, which are not very significant for a comparison.

MACS was tuned on the three-impulse case. In particular, the crowding factor $w_c$, the threshold $\rho_{\text{tol}}$ and the convergence $\rho_{\min}$ were kept constant to 1e−5, which is below the required local convergence accuracy, while $f_e$ and $n_{\text{pop}}$ were changed. A value of 1e-5 is expected to provide good local convergence and good density of the samples belonging to the Pareto front. Table 5 reports the value of performance indexes $p_{\text{conv}}$ and $p_{\text{spr}}$ over 200 runs of MACS with different settings. The index $p_{\text{spr}}$ and the index $p_{\text{conv}}$ have different, almost opposite, trends. However, it was decided to select the setting that provides the best convergence,i.e. $n_{\text{pop}} = 15$ and $f_e = 1/3$. This setting will be used for all the tests in this article.

**Table 5** Indexes $p_{\text{conv}}$ and $p_{\text{spr}}$ for different settings of MACS on the three-impulse case

| | $n_{pop}(\%) = 5$ | $n_{pop}(\%) = 10$ | $n_{pop}(\%) = 15$ |
|---|---|---|---|
| $p_{\text{conv}}$ | | | |
| $f_e = 1/3$ | 45.5 | 55.5 | 61.0 |
| $f_e = 1/2$ | 48.0 | 51.0 | 55.5 |
| $f_e = 2/3$ | 45.0 | 52.5 | 43.0 |
| $p_{\text{spr}}$ | | | |
| $f_e = 1/3$ | 68.5 | 62.0 | 56.0 |
| $f_e = 1/2$ | 65.0 | 57.0 | 46.0 |
| $f_e = 2/3$ | 67.5 | 51.0 | 36.5 |

On top of the complete algorithm, two variants of MACS were tested: one without individualistic moves and local archive, denoted as *no local* in the tables, and one with no attraction towards the global archive $A_g$, denoted as *no att* in the tables. Only these two variants are tested on these cases as they displayed the most significant impact in the previous standard test cases and more importantly were designed specifically to improve performances.

NSGA-II, PAES, MOPSO, and MTS were tuned as well on the three-impulse case. In particular, for NSGA-II the best result was obtained for 150 individuals and can be found in Table 6. A similar result could be obtained for MOPSO (Table 7). For MTS, only the population was changed while the number of individuals performing local moves was kept constant to 5. The results of the tuning of MTS can be found in Table 8. For the tuning of PAES the results can be found in Table 9.

All the parameters tuned in the three-impulse case were kept constant except for the population size of NSGA-II and MOPSO. The size of the population of NSGA-II and MOPSO was set to 100 and 40, respectively, on the two impulse case and was increased with the number of function evaluations in the Cassini case. In particular, for NSGA-II, the following ratios between population size and number of function evaluations were used: 272/180 000, 353/300 000, and 500/600 000. For MOPSO, the following ratios between population size and number of function evaluations were used: 224/180 000, 447/300 000, and 665/600 000. This might not be the best way to set the population size for these two algorithms, but it

**Table 6** NSGAII tuning on the three-impulse case

| Mean $M_{\text{conv}}$ | | | | Var $M_{\text{conv}}$ | | | |
|---|---|---|---|---|---|---|---|
| $\eta_c/\ \eta_m$ | 5 | 25 | 50 | $\eta_c/\ \eta_m$ | 5 | 25 | 50 |
| 5 | 36.1 | 38.3 | 43.0 | 5 | 201.0 | 202.0 | 185.0 |
| 10 | 32.3 | 39.4 | 40.6 | 10 | 182.0 | 172.0 | 182.0 |
| 20 | 31.7 | 39.6 | 42.5 | 20 | 175.0 | 183.0 | 169.0 |
| Mean $M_{\text{spr}}$ | | | | Var $M_{\text{spr}}$ | | | |
| $\eta_c/\ \eta_m$ | 5 | 25 | 50 | $\eta_c/\ \eta_m$ | 5 | 25 | 50 |
| 5 | 6.77 | 7.24 | 8.08 | 5 | 9.97 | 9.47 | 7.25 |
| 10 | 5.91 | 7.50 | 7.81 | 10 | 9.74 | 8.68 | 8.34 |
| 20 | 5.78 | 7.50 | 8.16 | 20 | 9.75 | 8.53 | 8.04 |
| $p_{\text{conv}}$ | | | | $p_{\text{spr}}$ | | | |
| $\eta_c/\ \eta_m$ | 5 | 25 | 50 | $\eta_c/\ \eta_m$ | 5 | 25 | 50 |
| 5 | 0.0% | 0.0% | 0.0% | 5 | 44.8% | 37.7% | 23.4% |
| 10 | 0.0% | 0.0% | 0.0% | 10 | 57.8% | 33.1% | 29.2% |
| 20 | 0.0% | 0.0% | 0.0% | 20 | 61.0% | 32.5% | 24.7% |

**Table 7** MOPSO tuning on the three-impulse case

| Mean $M_{conv}$ | | | | Var $M_{conv}$ | | | |
|---|---|---|---|---|---|---|---|
| Particles/subdivisions | 10 | 30 | 50 | Particles/subdivisions | 10 | 30 | 50 |
| 50 | 59.1 | 50.2 | 41.5 | 50 | 1080.0 | 1010.2 | 713.3 |
| 100 | 50.0 | 43.3 | 41.3 | 100 | 591.0 | 721.1 | 778.1 |
| 150 | 47.8 | 41.4 | 39.4 | 150 | 562.1 | 550.2 | 608.2 |
| Mean $M_{spr}$ | | | | Var $M_{spr}$ | | | |
| Particles/subdivisions | 10 | 30 | 50 | Particles/subdivisions | 10 | 30 | 50 |
| 50 | 16.1 | 13.5 | 12.3 | 50 | 41.3 | 37.3 | 24.0 |
| 100 | 14.7 | 12.2 | 11.6 | 100 | 32.8 | 25.8 | 24.5 |
| 150 | 14.8 | 11.9 | 11.4 | 150 | 30.0 | 22.2 | 22.5 |
| $p_{conv}$ | | | | $p_{spr}$ | | | |
| Particles/subdivisions | 10 | 30 | 50 | Particles/subdivisions | 10 | 30 | 50 |
| 50 | 0% | 0% | 0% | 50 | 0.5% | 2.5% | 3.5% |
| 100 | 0% | 0% | 0% | 100 | 0.5% | 4.5% | 4% |
| 150 | 0% | 0% | 0% | 150 | 0.5% | 2% | 4% |

**Table 8** MTS tuning on the three-impulse

| 3imp | Population | 20 | 40 | 80 |
|---|---|---|---|---|
| $M_{conv}$ | Mean | 17.8 | 22.6 | 23.6 |
| | Var | 97.6 | 87.8 | 73.2 |
| $M_{spr}$ | Mean | 12.6 | 19.9 | 18.4 |
| | Var | 34.7 | 26.2 | 18.6 |
| | $p_{conv}$ | 1.0% | 0.0% | 0.0% |
| | $p_{spr}$ | 0.5% | 0.0% | 0.0% |

**Table 9** PAES tuning on the three-impulse case

| Mean $M_{conv}$ | | | | Var $M_{conv}$ | | | |
|---|---|---|---|---|---|---|---|
| Subdivisions/mutation | 0.6 | 0.8 | 0.9 | Subdivisions/mutation | 0.6 | 0.8 | 0.9 |
| 1 | 53.7 | 70.6 | 70.2 | 1 | 525.0 | 275.0 | 297.0 |
| 2 | 52.8 | 70.2 | 70.0 | 2 | 479.0 | 266.0 | 305.0 |
| 4 | 53.0 | 70.2 | 70.1 | 4 | 453.0 | 266.0 | 311.0 |
| Mean $M_{spr}$ | | | | Var $M_{spr}$ | | | |
| Subdivisions/mutation | 0.6 | 0.8 | 0.9 | Subdivisions/mutation | 0.6 | 0.8 | 0.9 |
| 1 | 14.2 | 27.7 | 36.7 | 1 | 20.0 | 14.3 | 17.3 |
| 2 | 13.6 | 27.6 | 36.6 | 2 | 17.5 | 14.6 | 16.8 |
| 4 | 13.8 | 27.7 | 36.6 | 4 | 17.2 | 15.8 | 17.0 |
| $p_{conv}$ | | | | $p_{spr}$ | | | |
| Subdivisions/mutation | 0.6 | 0.8 | 0.9 | Subdivisions/mutation | 0.6 | 0.8 | 0.9 |
| 1 | 0.0% | 0.0% | 0.0% | 1 | 0.0% | 0.0% | 0.0% |
| 2 | 0.0% | 0.0% | 0.0% | 2 | 0.0% | 0.0% | 0.0% |
| 4 | 0.0% | 0.0% | 0.0% | 4 | 0.0% | 0.0% | 0.0% |

is the one that provided better performance in these tests.

Note that the size of the global archive for MACS was constrained to be lower than the size of the population of NSGA-II, in order to avoid any bias in the computation of $M_{spr}$.

The performance of all the algorithms on the Cassini case can be found in Table 10 for a variable number of function evaluations.

For the three-impulse case, MACS was able to identify an extended Pareto front (Figs 2(a) and (b)), where all the non-dominated solutions from all the 200 runs are compared to the global front), compared to the results in reference [**7**]. The gap in the Pareto front is probably due to a limited spreading of the solutions in that region. Note the cusp due the transition between the condition in which two-impulse solutions are optimal and the condition in which three-impulse solutions are optimal.

Table 10 summarizes the results of all the tested algorithms on the two-impulse case. The average value of the performance metrics is reported with, in parentheses, the associated variance over 200 runs. The two-impulse case is quite easy and all the algorithms have no problems identifying the front. However, MACS displays a better convergence than the other algorithms, while the spreading of MOPSO and MTS is superior to the one of MACS.

The three-impulse case is instead more problematic (Table 11). NSGA-II is not able to converge to the upper-left part of the front and therefore the convergence is 0 and the spreading is comparable to the one

of MACS. All the other algorithms perform poorly with a value of almost 0 for the performance indexes.

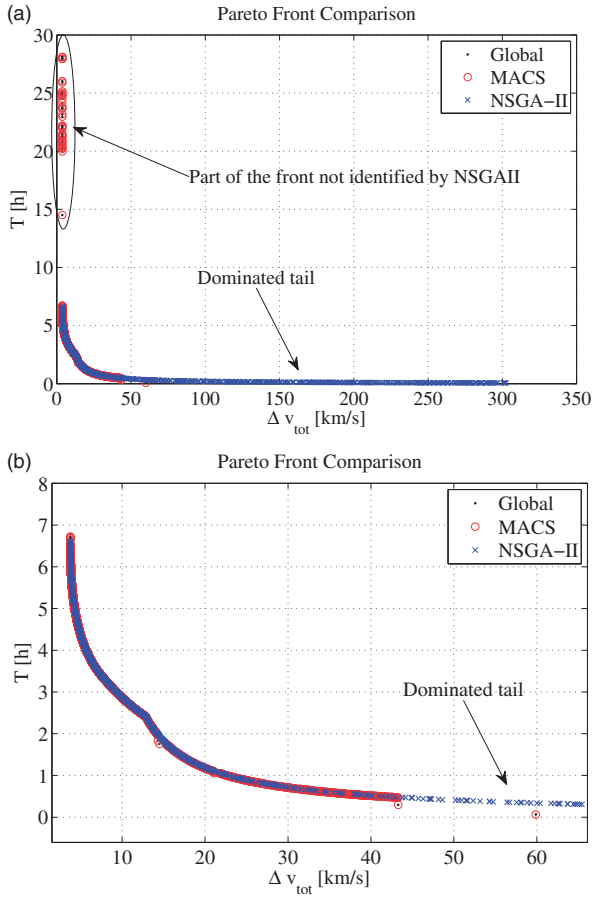


**Fig. 2** Three-impulse test case: (a) complete Pareto front, (b) close-up of the Pareto fronts

This is mainly due to the fact that no one can identify the vertical part of the front.

Note that the long tail identified by NSGA-II is actually dominated by two points belonging to the global front (Fig. 2(b)).

Table 12 reports the statistics for the Cassini problem. On top of the performance of the three variants tested on the other two problems, the table reports also the result for 10 agents and $f_e = 5$. For all numbers of function evaluations, MACS has better spreading than NSGA-II, because NSGA-II converges to a local Pareto front. Nonetheless, NSGA-II displays a more regular behaviour and a better convergence for low number of function evaluations although it never reaches the best front. The Pareto fronts are represented in Figs 3(a) and (b) (even in this case, the figures represent all the non-dominated solutions coming from all the 200 runs). Note that, the minimum $f$ returned by MACS is the best known solution for the single objective version of this problem [**9**]. All the other algorithms perform quite poorly on this case.

Of all the variants of MACS tested on these problems, the full complete one is performing the best. As expected, in all three cases, removing the individualistic moves severely penalizes both convergence and spreading. It is interesting to note that removing the attraction towards the global front is also comparatively bad. On the two-impulse case, it does not impact the spreading but reduces the convergence, while on the Cassini case, it reduces mean and variance but the success rates are zero. The observable reason is that MACS converges slower but more uniformly to a local Pareto front.

Finally, it should be noted that mean and variance seem not to capture the actual performance of the

**Table 10** Metrics $M_{\text{conv}}$ and $M_{\text{spr}}$, and associated performance indexes $p_{\text{conv}}$ and $p_{\text{spr}}$, on the two impulse test cases

| Metric | MACS | MACS no local | MACS no att | NSGA-II | PAES | MOPSO | MTS |
|---|---|---|---|---|---|---|---|
| $M_{\text{conv}}$ | 0.0077 | 0.039 | 0.0534 | 0.283 | 0.198 | 0.378 | 0.151 |
| | (1e-3) | (1.4e-2) | (9.5e-3) | (4.35e-3) | (0.332) | (0.0636) | (0.083) |
| $M_{\text{spr}}$ | 2.89 | 6.87 | 3.08 | 2.47 | 332.0 | 2.11 | 1.95 |
| | (0.49) | (7.36) | 0.943 | (0.119) | (2.61e4) | (2.65) | (1.41) |
| $p_{\text{conv}}$ | 98.5% | 91.5% | 84% | 0% | 75.5% | 9% | 57.5% |
| $p_{\text{spr}}$ | 29.5% | 0% | 24% | 62.5% | 0.5% | 94.5% | 85.5% |

**Table 11** Summary of metrics $M_{\text{conv}}$ and $M_{\text{spr}}$, and associated performance indexes $p_{\text{conv}}$ and $p_{\text{spr}}$, on the three impulse test cases

| Metric | MACS | MACS no local | MACS no att | NSGA-II | PAES | MOPSO | MTS |
|---|---|---|---|---|---|---|---|
| $M_{\text{conv}}$ | 5.53 | 7.58 | 154.7 | 31.7 | 53.0 | 39.4 | 17.8 |
| | (15.1) | (26.3) | (235.0) | (175.0) | (453.0) | (608.1) | (97.6) |
| $M_{\text{spr}}$ | 5.25 | 6.03 | 9.16 | 5.78 | 13.8 | 11.4 | 12.6 |
| | (3.73) | (3.95) | (2.07) | (9.75) | (17.2) | (22.5) | (34.7) |
| $p_{\text{conv}}$ | 61.0% | 40.5% | 0.0% | 0.0% | 0.0% | 0% | 1.0% |
| $p_{\text{spr}}$ | 56.0% | 36% | 0.0% | 61.0% | 0.0% | 4.0% | 0.5% |

**Table 12** Metrics $M_{conv}$ and $M_{spr}$, and associated performance indexes $p_{conv}$ and $p_{spr}$, on the Cassini case

| Approach | Metric | 180k | 300k | 600k |
|---|---|---|---|---|
| MACS 5/15 | $M_{conv}$ | 6.50 (229.1) | 4.48 (107.1) | 3.91 (62.6) |
| | $M_{spr}$ | 12.7 (126.0) | 11.1 (83.1) | 8.64 (35.5) |
| | $p_{conv}$ | 6% | 14% | 21.5% |
| | $p_{spr}$ | 27.5% | 31% | 41% |
| MACS 5/10 | $M_{conv}$ | 6.74 (217.9) | 5.56 (136.4) | 3.14 (50.1) |
| | $M_{spr}$ | 12.1 (83.0) | 10.3 (63.7) | 8.11 (30.0) |
| | $p_{conv}$ | 8% | 10% | 25.5% |
| | $p_{spr}$ | 26.0% | 31.5% | 45.5% |
| MACS no local | $M_{conv}$ | 13.5 (436.2) | 10.2 (350.1) | 7.86 (190.2) |
| | $M_{spr}$ | 31.1 (274.3) | 27.9 (278.9) | 21.9 (226.7) |
| | $p_{conv}$ | 1.0% | 1.5% | 2.5% |
| | $p_{spr}$ | 1.0% | 2.5% | 6% |
| MACS no att | $M_{conv}$ | 2.62 (1.46) | 2.2 (0.88) | 1.82 (0.478) |
| | $M_{spr}$ | 27.8 (83.2) | 22.9 (58.0) | 18.2 (28.0) |
| | $p_{conv}$ | 0.0% | 0.0% | 1.0% |
| | $p_{spr}$ | 0.0% | 0.0% | 0.0% |
| NSGA-II | $M_{conv}$ | 2.43 (18.0) | 1.99 (16.8) | 1.24 (1.62) |
| | $M_{spr}$ | 11.6 (71.4) | 11.0 (47.5) | 8.78 (28.2) |
| | $p_{conv}$ | 17.5% | 24.0% | 29.0% |
| | $p_{spr}$ | 15.5% | 12.5% | 25.0% |
| MOPSO | $M_{conv}$ | 2.62 (7.33) | 2.4 (2.57) | 2.14 (0.94) |
| | $M_{spr}$ | 28.0 (308.3) | 24.6 (260.4) | 21.8 (231.3) |
| | $p_{conv}$ | 0.5% | 1.0% | 1.0% |
| | $p_{spr}$ | 0.0% | 0.0% | 0.5% |
| PAES | $M_{conv}$ | 24.0 (54.5) | 19.8 (32.9) | 15.2 (16.6) |
| | $M_{spr}$ | 30.1 (47.3) | 26.0 (33.9) | 21.4 (19.5) |
| | $p_{conv}$ | 0.0% | 0.0% | 0.0% |
| | $p_{spr}$ | 0.0% | 0.0% | 0.0% |
| MTS | $M_{conv}$ | 3.71 (1.53) | 3.39 (1.67) | 3.02 (1.69) |
| | $M_{spr}$ | 18.1 (18.2) | 15.6 (13.4) | 13.1 (8.46) |
| | $p_{conv}$ | 0.0% | 0.0% | 0.0% |
| | $p_{spr}$ | 0.0% | 0.0% | 0.0% |

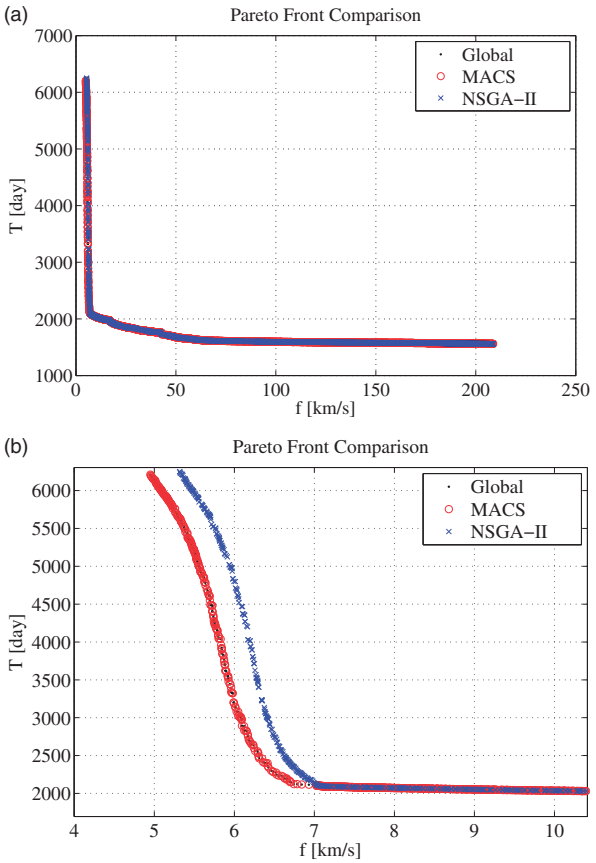


**Fig. 3** Cassini test case: (a) complete Pareto front and (b) close-up of the Pareto fronts

algorithms. In particular, they do not capture the ability to identify the whole Pareto front as the success rates instead do.

## 5. CONCLUSIONS

In this article, we presented a hybrid evolutionary algorithm for multi-objective optimization problems. The effectiveness of the hybrid algorithm, implemented in a code called MACS, was demonstrated at first on a set of standard problems and then its performance was compared against NSGA-II, PAES, MOPSO, and MTS on three space trajectory design problems. The results are encouraging as, for the same computational effort (measured in number of function evaluations, MACS was converging more accurately than NSGA-II on the two-impulse case and managed to find a previously undiscovered part of the Pareto front of the three-impulse case. As a consequence, on the three-impulse case, MACS, has better performance metrics than the other algorithms. On the Cassini case, NSGA-II appears to converge better to some parts of the front although MACS yielded solutions with better $f$ and identifies once more a part of the front that NSGA-II cannot attain. PAES and MTS do not perform well on the Cassini case, while MOPSO converges well locally but, with the settings employed in this study, yielded a very poor spreading.

From the experimental tests in this article, we can argue that the following mechanisms seem to be particularly effective: the use of individual local actions with a local archive as they allow the individuals to move towards and within the Pareto set; the use of an attraction mechanism as it accelerates convergence.

Finally, it should be noted that all the algorithms tested in this study use the Pareto dominance as selection criterion. Different criteria, like the decomposition in scalar subproblems, can be equality implemented in MACS, without disrupting its working principles, and lead to different performance results.

## ACKNOWLEDGEMENTS

The authors would like to thank Dr Edmondo Minisci and Dr Guilio Avanzini for the two-and three-impulse cases and the helpful advice.

## REFERENCES


1 **Coverstone-Caroll, V., Hartmann, J. W.,** and **Mason, W. M.** Optimal multi-objective low-thrust spacecraft trajectories. *Compu. Methods Appl. Mech. Eng.*, 2000, **186**, 387–402.

2 **Dachwald, B.** Optimization of interplanetary solar sailcraft trajectories using evolutionary neurocontrol. *J. Guid. Dyn.*, 2004, **27**, 66–72.

3 **Lee, S., von Allmen, P., Fink, W., Petropoulos, A. E.,** and **Terrile, R. J.** Multiobjective evolutionary algorithms for low-thrust orbit transfer optimization. In Proceedings of the Genetic and Evolutionary Computation Conference (GECCO 2005), Washington DC, USA, 25–29 June 2005.

4 **Schütze, O., Vasile, M. M.,** and **Coello Coello, C. A.** Approximate solutions in space mission design. In Parallel Problem Solving from Nature (PPSN 2008), Dortmund, Germany, 13–17 September, 2008.

5 **Schütze, O., Vasile, M., Junge, O., Dellnitz, M.,** and **Izzo, D.** Designing optimal low-thrust gravity-assist trajectories using space pruning and a multi-objective approach. *Eng. Optim.*, 2009, **41**(2), 155–181.

6 **Dellnitz, M., Ober-Blbaum, S., Post, M., Schtze, O.,** and **Thiere, B.** A multi-objective approach to the design of low thrust space trajectories using optimal control. *Celestial Mech. Dyn. Astron.*, 2009, **105**(1), 33–59.

7 **Minisci, E.** and **Avanzini, G.** Optimisation of orbit transfer manoeuvres as a test benchmark for evolutionary algorithms. In Proceedings of the 2009 IEEE Congress on Evolutionary Computation (CEC2009), Trondheim, Norway, May 2009, pp. 350–357.

8 **Vasile, M.** A behavioral-based meta-heuristic for robust global trajectory optimization. In Proceedings of the 2007 IEEE Congress on Evolutionary Computation (CEC2007), Singapore, September 2007, pp. 2056–2063.

9 **Vasile, M.** and **Locatelli, M.** A hybrid multiagent approach for global trajectory optimization. *J. Global Optim.*, 2009, **44**(4), 461–479.

10 **Adriana Lara, Gustavo Sanchez, Carlos A. Coello Coello,** and **Oliver Schütze**. HCS: A new local search strategy for memetic multi-objective evolutionary algorithms. *IEEE Trans. Evol. Comput.*, 2010, **14**(1), 112–132.

11 **Zhang, Q.** and **Li, H.** Moea/d: A multi-objective evolutionary algorithm based on decomposition. *IEEE Trans. Evol. Comput.*, 2007, **11**(6), 712–731.

12 **Price, K. V., Storn, R. M.,** and **Lampinen, J. A.** *Differential evolution. A practical approach to global optimization*, 2005 (Natural Computing Series, Springer, Berlin).

13 **Vasile, M.** and **Zuiani, F.** A hybrid multiobjective optimization algorithm applied to space trajectory optimization. In Proceedings of the IEEE International Conference on Evolutionary Computation, Barcelona, Spain, July 2010, pp. 308–315.

14 **Deb, K. A., Pratap, A.,** and **Meyarivan, T.** Fast elitist multi-objective genetic algorithm: Nga-ii. Kangal report no. 200001, KanGAL, 2000.

15 **Coello, C.** and **Lechuga, M.** Mopso: A proposal for multiple objective particle swarm optimization. Technical report evocinv-01-2001., CINVESTAV, Instituto Politecnico Nacional Col, San Pedro Zacatenco, Mexico, 2001.

16 **Knowles, J. D.** and **Corne, D. W.** The pareto archived evolution strategy: A new baseline algorithm for pareto multi-objective optimisation. In Proceedings of the IEEE International Conference on Evolutionary computation, Washington DC, US, 1999, pp. 98–105.

17 **Tseng, L.-Y.** and **Chen, C.** Multiple trajectory search for multi-objective optimization. In Proceedings of the IEEE International Conference on Evolutionary computation, Singapore, 25–28 September 2007, pp. 3609–3616.

18 **Van Veldhuizen, D. A.** and **Lamont, G. B.** Evolutionary computation and convergence to a pareto front. In Late Breaking papers at the Genetic Programming, California, July 1998, pp. 221–228.

19 **Vasile, M., Minisci, E.,** and **Locatelli, M.** On testing global optimization algorithms for space trajectory design. In Proceedings of the AIAA/AAS Astrodynamic Specialists Conference Honolulu, Hawaii, USA, August 2008.

20 **Avanzini, G.** A simple Lambert algorithm. *J. Guidance, Control, Dynamics*, 2008, **31**(6), 1587–1594.

21 **Battin, R.** *An Introduction to the mathematics and methods of astrodynamics.*, 1999 (AIAA).

22 **Myatt, D. R., Becerra, V. M., Nasuto, S. J.,** and **Bishop, J. M.** *Global optimization tools for mission analysis and design.* 2004, Final rept. esa ariadna itt ao4532/18138/04/nl/mv,call03/4101, ESA/ESTEC.


## APPENDIX

### Notation

| | |
|---|---|
| $a_T$ | semimajor axis |
| $A_g$ | global archive |
| $A_l$ | local archive |
| $D$ | search space |
| $e_T$ | eccentricity |
| $f$ | cost function |
| $f_e$ | fraction of the population doing local moves |
| $I_d$ | dominance index |
| $M_{\text{conv}}$ | convergence metrics |
| $M_{\text{spr}}$ | spreading metrics |
| $n_{\text{eval}}$ | number of function evaluations |
| $n_{\text{pop}}$ | population size |
| $N_\rho$ | neighbourhood of solution $\mathbf{x}$ |
| $N_e$ | maximum number of allowed function evaluations |

| | | | |
|---|---|---|---|
| $p_{\text{conv}}$ | percentage of success on convergence | $\mathbf{u}$ | variation of the solution $\mathbf{x}$ |
| $p_{\text{spr}}$ | percentage of success on spreading | $U$ | uniform distribution |
| $P_i$ | $i$th planet | $w_c$ | tolerance on the maximum allowable crowding |
| $P_k$ | population at generation $k$ | | |
| $r$ | random number | $\mathbf{x}$ | solution vector |
| $r_p$ | pericentre radius | $X$ | Pareto optimal set |
| $r_{\text{pmin}}$ | minimum pericentre radius | $\mathbf{y}$ | mutate individual |
| $s$ | resource index | | |
| $S$ | selection function | $\Delta v$ | variation of velocity |
| $t_0$ | departure time | $\theta_i$ | true anomaly of manoeuvre $i$ |
| $t_i$ | manoeuvre time | $\mu_E$ | gravity constant |
| $t_f$ | final time | $\rho$ | size of the neighbourhood $N_\rho$ |
| $T$ | transfer time | | |