Improved Magnetic Charged System Search Optimization Algorithm with Application to Satellite Formation Flying

Andrea D'Ambrosio[a,*], Dario Spiller[b], Fabio Curti[c]

[a] *PhD student, School of Aerospace Engineering, La Sapienza University of Rome, Via Salaria 851*
[b] *PostDoc, School of Aerospace Engineering, La Sapienza University of Rome, Via Salaria 851*
[c] *Associate Professor, School of Aerospace Engineering, La Sapienza University of Rome, Via Salaria 851*

Abstract

This paper is devoted to the implementation and application of an improved version of the metaheuristic algorithm called magnetic charged system search. Some modifications and novelties are introduced and tested. Firstly, the authors' attempt is to develop a self-adaptive and user-friendly algorithm which can automatically set all the preliminary parameters (such as the numbers of particles, the maximum iterations number) and the internal coefficients. Indeed, some mathematical laws are proposed to set the parameters and many coefficients can dynamically change during the optimization process based on the verification of internal conditions. Secondly, some strategies are suggested to enhance the performances of the proposed algorithm. A chaotic local search is introduced to improve the global best particle of each iteration by exploiting the features of ergodicity and randomness. Moreover, a novel technique is proposed to handle bad-defined boundaries; in fact, the possibility to self-enlarge the boundaries of the optimization variables is considered, allowing to achieve the global optimum even if it is located on the boundaries or outside. The algorithm is tested through some benchmark functions and engineering de-

*Corresponding author
Email addresses: andrea.dambrosio@uniroma1.it (Andrea D'Ambrosio), dario.spiller@uniroma1.it (Dario Spiller), fabio.curti@uniroma1.it (Fabio Curti)

sign problems, showing good results, followed by an application regarding the problem of time-suboptimal manoeuvres for satellite formation flying, where the inverse dynamics technique, together with the B-splines, is employed. This analysis proves the ability of the proposed algorithm to optimize control problems related to space engineering, obtaining better results with respect to more common and used algorithms in literature.

Keywords: metaheuristic physics-inspired algorithm, constrained optimization, trajectory planning, satellite formation flying

---

Nomenclature

CP = Charged Particle

CLS = Chaotic Local Search

N_{CP} = Number of CPs

D = Number of the variables to optimize (dimension of each CP)

W = Maximum order of magnitude of the problem

$\mathbf{UB}, \mathbf{LB}$ = Upper and lower bound vector

r = Real uniformly-distributed random number within the range [0,1]

r_{in} = Integer uniformly-distributed random number

$G_{(\cdot)}$ = Maximum number of iterations

$\mathbf{X}$ = Position vector

$\mathbf{V}$ = Velocity vector

$\mathbf{F}$ = Force vector

q = Charge associated to each CP

I = Electric current associated to each CP

r_{ij} = Separation distance between CPs

ϵ = Small positive constant

δ = Tolerance

p_{ij} = Probability of attraction/repulsion between CPs

pm_{ij} = Probability of the ith wire (CP) magnetic influence on the jth CP

p_r = Probability that an electrical force is a repelling force

$w_{(\cdot)}$, $z_{(\cdot)}$ = Electric and magnetic coefficients equal to 0 or 1

Z = Coefficient of the chaotic local search

CM = Charged Memory (contain the best CPs)

CMS = Charged Memory Size

$N_{CP,ar}$ = Number of attracting/repelling CPs

$k_{v/a}$ = Coefficient of the velocity/acceleration term

k_{ar} = Coefficient of attraction/repulsion

J = Performance index

S = Entropy of the system

C_{max} = Maximum number of allowed boundaries violations

$c_{(\cdot)}$ = Generic counter

$\boldsymbol{a}$, $\boldsymbol{b}$ = Approximation coefficient vectors

$\mathcal{B}$ = B-spline approximation polynomial

$\mathscr{I}$ = Coordinate system associated to Earth-centered inertial reference frame

r_c = Radius of chief orbit, km

i_c = Inclination of chief orbit, rad

$\mathbf{K}$ = B-spline knot vector

$\mathscr{L}$ = Coordinate system of the local-vertical/local-horizontal reference frame

$\mathcal{N}_{i,j}$ = B-spline basis functions

$N_{\mathcal{P}}$ = Number of interpolation parameters

$\mathbf{r}$ = Inertial position, km

$\mathbf{u}$ = External acceleration, km/s^2

$\mathbf{x}$ = State vector

μ = Earth gravitational constant, km^3/s^2

$\mathbf{y}$ = Flat output

$\boldsymbol{\rho}, \dot{\boldsymbol{\rho}}, \ddot{\boldsymbol{\rho}}$ = Relative position (km), velocity (km/s), acceleration (km/s^2) vectors

ω = Angular velocity, rad/s

Subscripts and superscripts

g = Index associated to the global best CP

k = Index associated to the k-th internal iteration

k_f = Index associated to the last internal iteration

K = Index associated to the K-th external iteration

E = Subscript referring to electric force/acceleration

B = Subscript referring to magnetic force/acceleration

N = Numerical approximation of functions or curves

$(0, f)$ = Quantity referring to beginning/end of manoeuvre

(c, d) = Quantity referring to chief/deputy satellite

1. Introduction

Optimization is a widely spread procedure which consists in finding the optimal solution of a given problem among all the admissible ones. Nowadays, this procedure is exploited in many fields, such as physics (Hartmann & Rieger, 2003), economics (Mohajan et al., 1953) and engineering (Kalyanmoy, 2004). Complex optimization problems are solved thanks to computing algorithms, which are divided into deterministic and stochastic categories. Deterministic algorithms follow a rigorous and repeatable procedure and for the same starting point the final result does not change. On the other hand, stochastic algorithms, which include metaheuristics, are characterized by random processes that cause the path not to be always the same with the same initial conditions. Thus, the final solutions could be different each time the algorithm is launched (Yang, 2010), but it is expected that some of them are nearly optimal, even though there is no guarantee for such optimality (Yang, 2018). In addition, the randomization allows to move from local to global search; therefore, they are suitable for global optimization.

In the recent years, metaheuristic algorithms have been more and more exploited for optimization problems because of their simplicity and advantages. The computational costs have been compensated by the fast development of the computing technology. Many metaheuristic algorithms have been proposed; they can be subdivided into two main groups: bio-inspired and physics-inspired algorithms. The first ones are related to animal biology and behaviour, whereas the second ones are based on physics laws. Among the most employed algorithms there are: Genetic Algorithm (GA) (Holland, 1975; Goldberg, 1989),

inspired by the Darwin's theory of evolution; Ant Colony Optimization (ACO) (Dorigo et al., 1996), inspired by the behaviour of ants; Particle Swarm Optimization (PSO) (Clerc, 2010), based on the social behaviour of human kind and on swarms of birds; Differential Evolution (DE) (Storn & Price, 1997), based on the concepts of mutation, recombination and selection applied to a population; and many others (Bandaru & Deb, 2016).

Among these, Charged System Search (CSS) (Kaveh & Talatahari, 2010b) and its upgraded version Magnetic Charged System Search (MCSS) (Kaveh et al., 2013) exploit respectively the electric and the electro-magnetic forces and the uniformly accelerated motion laws to make the particles move through the domain. Considering the good performances of MCSS, this algorithm has been chosen to be further developed in this work. Until now, MCSS has mainly been used for structural design optimization (Kaveh & Zolghadr, 2014; Kaveh et al., 2018), although there is also another work (Kumar et al., 2016) that applies successfully the MCSS to a more common topic in artificial intelligence, which is the clustering optimization (a very useful technique used for example in machine learning and pattern recognition); in particular, in (Kaveh et al., 2014, 2015) an improved version of MCSS has already been proposed by the authors, but the reported improvements are only related to the part of the algorithm dealing with the harmony search, whereas the improvements proposed in the present paper concern the whole structure of the algorithm. Indeed, this paper deals with the development of an Improved version of the Magnetic Charged System Search algorithm, called IMCSS, and its applications in space trajectory planning, reagrading in particular satellite formation flying manoeuvres.

A problem associated to metaheuristic algorithms, including MCSS, is the proper setting of some internal parameters as the number of maximum iterations, the population size and the internal coefficients. These parameters may change according to the different problems that need to be optimized, since they can affect significantly the results of the optimization procedure. Thus, this poses a limitation in the usage of this kind of algorithms, which are restricted only to expert users in this field. Some works dealing with the self-adaptivity of

metaheuristic algorithms, i.e., the ability to automatically tune the internal parameters, have been already developed. For example, many self-adaptive PSO have been proposed in literature and some of them have been analysed in (Harrison et al., 2017) to explore their effectiveness and convergence. A self-adaptive version of the Gravitational Search Algorithm (GSA) (Rashedi et al., 2009), with a modified chaotic local search, has been introduced in (Ji et al., 2017). Civicioglu et al. (Civicioglu et al., 2018) presented a weighted DE (WDE), in which all the internal parameters are determined randomly, so it has no control parameters but the pattern size. Moreover, many tuning approaches have been investigated in order to find the optimal values of the internal parameters of metaheuristic algorithms, such as F-Race (Birattari et al., 2002), Revac (Nannen & Eiben, 2007), SPO (Bartz-Beielstein et al., 2005), CRS-Tuning (Vecek et al., 2016).

Considering what has been said so far, the main contributions of this paper are as follows:

1. The new algorithm is able to automatically determine i) a suitable number of particles the population is made up of, ii) the maximum iterations number, and iii) the internal parameters which regulate all the phases of the algorithm. Hence, the user can apply this optimization algorithm as it was a "black box", i.e., without knowing anything about it, just by defining the search space (lower and upper bounds of the variables).
2. Some strategies are proposed to make the algorithm more efficient.
   (a) A chaotic local search is introduced to improve the best particle of the current iteration, by exploring its local neighborhood, and thus helping the convergence.
   (b) The algorithm is built in order to have external loops and internal loops, where the first ones increase the ability of exploration and the second ones exploit the fast convergence of the original algorithm.
   (c) A method to compute a sort of "entropy" of the system is provided by considering the objective functions of the whole population. Thanks

to this information, the algorithm can understand if it is converging to an optimal solution, meaning that most of the particles approach the neighbourhood of the global best particle, or if the convergence is difficult to reach. Therefore, if needed, the population size can be self-increased (for the following external loops) to help the optimization procedure be successful.

(d) The proposed algorithm also takes into account the possibility to have a bad-defined search space. Indeed, the algorithm can autonomously enlarge the boundaries proposed by the user and catch minima excluded by the original domain required for the initialization.

3. At the best of the author's knowledge, this algorithm has never been used before for space applications. Thus, this paper shows that IMCSS performs well to optimize very challenging problems in space engineering. In order to justify the optimality of the obtained results, a comparison with previous papers is carried out.

The space application reported in this work consists of a reconfiguration manoeuvre for satellite formation flying. Metaheuristic algorithms are used for space engineering optimization; in particular, they have been used to face the problem of guidance, i.e., how to plan spacecraft optimal trajectories. For example, PSO has been used to optimize finite-thrust rendezvous trajectories (Pontani & Conway, 2013) and to exploit attitude control to minimize the reconfiguration time for satellite formations (Spiller et al., 2017); a genetic algorithm has been employed to obtain optimal rendezvous manouevres in (Kim & Spencer, 2002) and (Zhang et al., 2008); the evolutionary computation (EC) has been used in (Wang & Zheng, 2012) for satellite formation reconfiguration purposes. In order to successfully face the optimal control problem, the inverse dynamics approach, already employed for trajectories optimization (Ross & Fahroo, 2002; Boyarko et al., 2011; Spiller et al., 2016a,b), is considered in this research along with the differential flatness approach (Fliess et al., 1995; Louembet, 2007). As a consequence, state and control variables are expressed

as a function of a minimum number of optimization parameters named flat outputs. For example, for satellite formation manoeuvres the flat outputs can be well-represented by the parameters describing the relative displacements between the satellites. The advantage of using this approach is that the dynamical constraints are satisfied a priori and the flat output can be chosen to automatically fulfil the boundary constraints. Moreover, the control policy is obtained in an analytical closed form. In particular, B-spline curves are chosen to describe the fat outputs because they provide very good results using a minimal number of support functions (Boor, 1972).

This paper is organized as follows. Sec. 2 describes the features of the proposed algorithm IMCSS. In Sec. 3, IMCSS is used to find the global minima of some benchmark functions and well-known engineering optimization problems in order to test the effectiveness and the convergence of the algorithm. The performances of the proposed algorithm are also compared to other works in literature. Sec. 4 reports the application of IMCSS for the space guidance problem concerning relative motion. The used reference frames and dynamical model are provided along with the main characteristics of the inverse dynamics technique and the B-spline curves. Sec. 5 deals with the numerical results of the optimization and, finally, concluding remarks are given in Sec. 6.

2. Improved Magnetic Charged System Search algorithm

In this Section, the IMCSS is introduced. This algorithm is an improved version of the metaheuristic magnetic charged system search (MCSS) algorithm, presented in (Kaveh et al., 2013), where it was demonstrated that the algorithm has a very good searching ability and convergence speed (especially when the number of particles and iterations is not so high). Generally, MCSS can be defined as a physics-inspired algorithm which exploits the electro-magnetic force and the Newton mechanics laws to make the charged particles (CPs) move through the search space.

It is noteworthy that all the parameters of the proposed algorithm, together

with their associated mathematical laws, are not intended to be certainly suitable for every problem, but they have performed very well in all the analysis carried out in this paper. Since different optimization problems are proposed and solved (various benchmark functions with diverse features, different enginnering design problems and the space guidance application), there is a good evidence that the parameters have been properly chosen.

2.1. Preliminary considerations

Before explaining in detail the algorithm, some preliminary considerations are introduced. One can associate to each CP a vector of variables to be optimized, called the position vector ($\mathbf{X}_i$), and a performance index (J_i), also referred to as objective function. Throughout this paper the subscript i (and also j) is used to indicate the $i(j)$-th CP and therefore it varies in the range $[1, N_{CP}]$, where N_{CP} is the number of CPs. The length of the vector $\mathbf{X}_i$ represents the dimension of the problem, called D, whereas W indicates the maximum order of magnitude of the problem. This value is computed as the order of magnitude of the maximum value among the components of the difference vector between the upper bound (**UB**) and the lower bound (**LB**) vectors, i.e.

$$W = |\lfloor \log_{10}(\max(\mathbf{UB} - \mathbf{LB})) \rfloor| \,, \tag{1}$$

where $\lfloor \rfloor$ indicates the nearest integer towards minus infinity and $| |$ the absolute value. The idea of IMCSS is to link the number of CPs and the maximum iterations numbers to the only paramters defined by the (non-expert) user, i.e., **UB** and **LB**. By knowing W, it is possible to compute the number of CPs associated to the problem

$$N_{CP} = 10 \cdot \{W + r_{in}[2,\ 3\lceil \log(D+1) \rceil]\} \tag{2}$$

In Eq. (2), $\lceil \rceil$ and r_{in} represent respectively the nearest integer towards infinity and a random integer number between the two in brackets. Obviously, a maximum number of CPs must be set in order not to have huge numbers of CPs

which would make the computation too slow. Thus, the maximum number of CPs ($N_{CP,max}$) has to be imposed. Generally, it is a free choice parameter and it can be decided by the user. In this algorithm, it is reasonably set equal to 50.

Metaheuristic algorithm can require many iterations to optimize very complex problems. The proposed algorithm is divided into external and internal loops. In the internal loop, IMCSS exploits the characteristic of the original MCSS, i.e., the fast convergence even for a low number of iterations. On the contrary, the external loops enhance the exploration ability of IMCSS, preserving at the same time the results of the previous iterations. For what concerns with the maximum internal iterations number (G_k), the following law is considered

$$G_k = 600 - 3N_{CP} \tag{3}$$

Also in this case, a maximum number of internal iterations is implicitly set for the same aforementioned reason. The choices behind Eqs. (2) and (3) are related to the fact that if the dimension increases, also the complexity of the problem increases and thus a greater number of CPs could be required for the success of the algorithm.

In addition, if the number of internal iterations was constant and the number of CPs was too small, the search space could not be explored well enough, and this is the reason why a decreasing law is chosen for G_k. If N_{CP} is low, G_k is high to compensate the reduced number of particles with the possibility to have more time (iterations) to explore the search space. On the contrary, if N_{CP} is high, the search space is better explored starting from the beginning of each internal loop, because the particles are more widely distributed.

The external iterations number (G_K) is defined as

$$G_K = \max(\lceil 12 - G_k/10^{\lfloor \log_{10}(G_k) \rfloor} \rceil \; ; \; 3), \tag{4}$$

where the function $10^{\lfloor \log_{10}(G_k) \rfloor}$ allows to extract the order of magnitude of G_k. Moreover, if G_k increases, G_K decreases, which means that if many internal

iterations have to be performed by the algorithm, it does not require many other external loops. In order to improve the convergence at the end of the algorthm, the number of iterations of the last internal cycle (G_{kf}) is increased, i.e.

$$G_{kf} = 5G_k. \tag{5}$$

2.2. Core of IMCSS

Once the numbers of iterations and CPs have been determined, it is possible to start with the actual optimization procedure. At the beginning of each external loop, each component of the position vector associated to each CP is initialized randomly within the range $[LB_m, UB_m]$, where the subscript m refers to the mth component of the particle, with $m \in [1, D]$. The initial velocity of all the CPs ($\mathbf{V}_i^0$) is set equal to the null vector. Afterwards, the performance index is assigned to each CP and the objective functions are sorted increasingly along with the corresponding CPs. Before starting a new external loop, the first re-initialized particle is substituted by the global best particle of the last external loop. This substitution allows the algorithm to have a continuity in the search of the optimized variables, although the algorithm is re-initialized G_K times. The first best CPs are stored in the Charged Memory (CM), and they are updated at each iteration. The number of the CPs stored in CM is defined as CMS (throughout this paper CMS is set equal to $N_{CP}/5$). Other information that must be stored and updated accordingly during the whole algorithm are the best and worst objective functions.

The charge q_i^k of each CP at the step k is computed by taking into account the current performance index of the CP (J_i), the best (J_g) and worst (J_{worst}) fitnesses among all the CPs so far, i.e.

$$q_i^k = \frac{J_i^k - J_{worst}}{J_g - J_{worst}}. \tag{6}$$

The charges must be stored because they are subsequently used to obtain the electric current. In this paper, the indices k and g are used to indicate re-

spectively the current iteration and the best CP, i.e., the CP with the best performance index. As it can be seen from Eq. (6), the values of the charges are normalized within the range $[0, 1]$, where 0 is associated to the worst CP and 1 to the best CP.

Supposing that CPs move in virtual straight wire of radius R_w, they create magnetic fields in the space, which generate magnetic forces on the other CPs. According to (Kaveh et al., 2013), the average electric current (I_{avg}) can be computed in two different ways,

$$(I_{avg})_i^k = \frac{q_i^k - q_i^{k-1}}{q_i^k + \epsilon}, \tag{7}$$

or

$$(I_{avg})_i^k = \text{sign}(J_i^k - J_i^{k-1}) \cdot \frac{df_i^k - df_{min}^k}{df_{max}^k - df_{min}^k}, \tag{8}$$

$$df_i^k = \left| J_i^k - J_i^{k-1} \right|. \tag{9}$$

In Eq. (7), ϵ defines a small positive value which is adopted to avoid singularity. In Eq. (8), the function df represents the absolute value of the objective function variation of the i-th CP in the k-th movement (iteration), the function sign$(\cdot)$ is the sign function, and df_{max}^k and df_{min}^k are respectively the maximum and the minimum values of df. Hereafter, based on the findings of Ref. (Kaveh et al., 2013), Eq. (8) is adopted. In order to obtain the electric and magnetic forces acting on all CPs for each iteration, the separation distance between all the CPs and the distance between the i-th wire and the j-th CP must be computed. In this work, these two distances are supposed to be equal and are indicated with r_{ij}. The explicit expression of r_{ij} is given by (Kaveh & Talatahari, 2010b; Kaveh et al., 2013)

$$r_{ij} = \frac{||\mathbf{X}_i - \mathbf{X}_j||}{||(\mathbf{X}_i + \mathbf{X}_j)/2 - \mathbf{X}_g|| + \epsilon} \tag{10}$$

As already stated, $\mathbf{X}_{i/j}$ represents the position of the generic CP (i.e., the vector of optimized variables) and ϵ is a small positive constant to prevent singularity.

Let us consider insulating solid spheres of radius R_a with a uniform volume charge density, each of which carrying a total charge q_i. The electric force acting on the j-th CP and produced by multiple CPs is

$$\mathbf{F}_{E,j} = k_e q_j \sum_{i,i\neq j} \left(\frac{q_i}{R_a^3} r_{ij} w_1 + \frac{q_i}{r_{ij}^2} w_2 \right) \frac{\mathbf{r}_i - \mathbf{r}_j}{||\mathbf{r}_i - \mathbf{r}_j||} \tag{11}$$

where

$$\begin{cases} w_1 = 1,\ w_2 = 0 \iff r_{ij} < R_a \\ w_1 = 0,\ w_2 = 1 \iff r_{ij} \geq R_a \end{cases}$$

As well-known from physics laws, there are two contributions within the brakets of the right-handed side of Eq. (11): the first one represents a linear relation with the distance r_{ij}, valid inside the sphere, and the second one is inversely proportional to the square of the distance, valid outside the sphere. In analogy with the real electric force, the equation is adapted for the case of this metaherustic algorithm in the folowing way

$$\mathbf{F}_{E,j} = q_j \sum_{i,i\neq j} \left(\frac{q_i}{R_a^3} r_{ij} w_1 + \frac{q_i}{r_{ij}^2} w_2 \right) \cdot p_{ij} \cdot (\mathbf{X}_i - \mathbf{X}_j) \tag{12}$$

where

$$\begin{cases} w_1 = 1,\ w_2 = 0 \iff r_{ij} < R_a \\ w_1 = 0,\ w_2 = 1 \iff r_{ij} \geq R_a \end{cases}$$

and

$$p_{ij} = \begin{cases} 1 & \text{if} \quad \frac{J_i - J_g}{J_j - J_i} > r \text{ or } J_j > J_i \\ 0 & \text{otherwise} \end{cases}$$

Here, p_{ij} indicates the probability of attraction of the i-th CP by the j-th CP and r represents a real uniformly-distributed random number within the range [0,1]. The radius of the insulating sphere R_a is set equal to 1 in this

work, although other possibilities can be considered, such as the one proposed in (Kaveh & Talatahari, 2010b)

$$R_a = 0.10 \cdot \max(\ \mathbf{X}_{i,max} - \mathbf{X}_{i,min} \mid i = 1, 2, ..., N_{CP}), \tag{13}$$

based on the size of the search space. For what concerns with the magnetic force acting on the jth CP, it is a linear function of r_{ij} inside the wire and it is inversely proportional to the square of r_{ij} outside, as it is known from physics. Thus, the magnetic force is

$$\mathbf{F}_{B,j} = q_j \sum_{i,i \neq j} \left(\frac{I_i}{R_w^3} r_{ij} z_1 + \frac{I_i}{r_{ij}} z_2 \right) \cdot pm_{ij} \cdot (\mathbf{X}_i - \mathbf{X}_j) \tag{14}$$

where

$$\begin{cases} z_1 = 1,\ z_2 = 0 \iff r_{ij} < R_w \\ z_1 = 0,\ z_2 = 1 \iff r_{ij} \geq R_w \end{cases}$$

and

$$pm_{ij} = \begin{cases} 1 & \text{if} \quad J_j > J_i \\ 0 & \text{otherwise} \end{cases}$$

In Eq. (14), R_w is set equal to 1 and pm_{ij} is the probability of the magnetic influence (attracting or repelling) of the i-th wire on the j-th CP. This factor is very important because only a good CP can influence a bad CP. Instead, this does not happen for the electric force, where good and bad CPs can influence each other. In addition, in order to emphasize the best CPs and give them much more importance in the algorithm, the number of "active" CPs (the CPs which actually attract/repel) is not assumed to be equal to N_{CP}, but only to a fraction of the whole population. This fraction can be a free parameter decided by the user and it can affect significantly the optimization procedure; thus, it must be set carefully. A reasonable choice for the active CPs, found during this work, is $N_{CP}/10$, which is the value used throughout this paper. The total force acting on the j-th CP is then

$$\mathbf{F}_j = p_r \mathbf{F}_{E,j} + \mathbf{F}_{B,j} \tag{15}$$

$$p_r = \begin{cases} 1 & \text{if} \quad r > k_{ar} \cdot (1 - k/G_k) \\ -1 & \text{otherwise} \end{cases} \tag{16}$$

where p_r is the probability that an electrical force is a repelling force. The expression $k_{ar} \cdot (1 - k/G_k)$ decreases with the iterations: this means that the repelling forces ensure the exploration at the beginning of the algorithm, whereas, at the end, the electric forces become more and more attractive, so that the exploitation is increased. The value of k_{ar} is defined as follows

$$k_{ar} = \min(G_k/1000 + 10/N_{CP} \; ; \; 0.5) \tag{17}$$

The reason behind the form of Eq. (17) is due to the fact that when G_k is high, there is more "time" (iterations) to explore the search space, thus it is possible to have a great repulsion (high value of k_{ar}). On the contrary, when G_k is not so high, the exploration phase is reduced in favor of a faster exploitation, in order to ensure the convergence. For what concerns with the parameter N_{CP}, if its value is high, the exploration is already ensured at the beginning by the wide diffusion of the CPs in the search space; thus, the exploitation is more encouraged (this is the reason why an inverse proportionality is chosen). The maximum admitted value for k_{ar} is equal to the probability of 50% between attraction and repulsion.

Once the forces are computed, the motion of the CPs has to be considered. Even in this case, the equations of motion are inspired by the physics laws, in particular by the uniformly accelerated motion laws.

$$\mathbf{X}_j^{k+1} = r_{j1} \cdot k_a \cdot \frac{\mathbf{F}_j}{m_j} \cdot \Delta t^2 + r_{j2} \cdot k_v \cdot \mathbf{V}_j^k \cdot \Delta t + \mathbf{X}_j^k \tag{18}$$

$$\mathbf{V}_j^{k+1} = \frac{\mathbf{X}_j^{k+1} - \mathbf{X}_j^k}{\Delta t} \tag{19}$$

In Eqs. (18) and (19), r_{j1} and r_{j2} are two random numbers that are uniformly distributed in the range [0,1] and they replace the real physics coefficients determined by the double integration of the acceleration (in particular, r_{j1} replaces 1/2 and r_{j2} replaces 1). In addition, the time interval Δt is set equal to the unity and the mass of each CP (m_j) is indeed substituted by its charge (q_j). This last choice makes the simplification of a fraction ($\mathbf{F}_j/q_j$) occur and allows to take into account also the electric and magnetic accelerations (Eqs. (12),(14)) acting on the worst CP (for which $q_j = 0$); otherwise, a null force would be applied on it. The remaining coefficients are respectively the coefficient of the acceleration (k_a) and the coefficient of the velocity (k_v). Their expressions are given by

$$k_a = k_{a,0} + (k_{a,f} - k_{a,0}) \cdot k/G_k \tag{20}$$

$$k_v = k_{v,f} + (k_{v,0} - k_{v,f}) \cdot k/G_k \tag{21}$$

As it can be seen from Eqs. (20) and (21), considering $k_{a/v,f} > k_{a/v,0}$, k_a is a linearly increasing function, while k_v is linearly decreasing. The initial and final values of k_a and k_f are chosen to be

$$\begin{cases} k_{v,f} = 1 + r \cdot \left(\dfrac{1}{N_{CP}} + \dfrac{G_k}{10^{\lceil \log_{10}(G_k) \rceil}} \right) \\ k_{v,0} = 0.8 \\ k_{a,0} = \lceil k_{v,f} \rceil - k_{v,f} \\ k_{a,f} = 2 \cdot k_{a,0} \end{cases} \tag{22}$$

Eq. (18) shows that the new position of each CPs is determined by three main contributions: the old position and the terms related to both the velocity and the acceleration. The term related to the velocity can be seen as the inertia term which appears in the velocity equation of the PSO (Clerc, 2010); thus k_v plays here the role that w plays in the PSO. It's important that the inertia (velocity) term is more relevant at the beginning of the algorithm to ensure the exploration, but then it should be decreased in favor of the exploitation. On the other hand, the acceleration term should be more relevant at the end of the iterations to improve the exploitation in the neighborhood of the best CP: this

idea can also be seen clearly from Eqs. (15) and (16), where the probability of attraction increases with the iteration number. These are the reasons why the terms k_a and k_v are chosen to be respectively increasing and decreasing with time. The inverse and direct proportionalities, with respect to N_{CP} and G_k, appearing in the first equation of 22, have basically the same explanation already provided for Eq. (17).

2.3. Chaotic local search

To enhance the performance of the IMCSS, a chaotic local search (CLS) is introduced. Chaotic search strategy is inspired by the chaos phenomenon in nature, which is a classic nonlinear dynamical system with the properties of ergodicity, randomness, and sensitivity to its initial conditions (Jia et al., 2011).Thanks to these features, a chaotic system can randomly generate a long-time sequence which is able to traverse through every state of the system without repetitions in certain ranges (Guo et al., 2015). In order to achieve a better balance between exploration and exploitation abilities of the IMCSS and improve the convergence of the whole algorithm, more heuristic information is integrated, such as the search direction provided by the best particles in the current population. Based on these considerations, a merge between the superior information of the current population and the chaotic search process is performed.

The chaotic local search employed for the proposed algorithm is inspired by the work of Ji et al. (Ji et al., 2017). Thus, the main features of their chaotic local search are employed, but with some differences. The chaotic local search is exploited in order to improve the best CP within the current iteration by searching in its neighborhood. This strategy could help to reach faster the global best. Therefore, the following criterion is applied

$$\mathbf{X}_{g1}^{k} = \mathbf{X}_{g}^{k} + (Z - 0.5) \cdot (\mathbf{X}_{CM_{r_{i1}}}^{k} - \mathbf{X}_{CM_{r_{i2}}}^{k}) \iff r_1^2 < r_2 \tag{23}$$

In Eq. (23), $\mathbf{X}_{g1}^{k}$ is a new possible global best, $\mathbf{X}_{CM_{r_{i1/2}}}^{k}$ are two random CPs chosen among the Charged Memory, and $r_{1/2}$ are two real random numbers. The condition $r_1^2 < r_2$ makes the local search occur with enough probability,

since $r_{1/2}$ varies within the range $[0, 1]$ and therefore r_1^2 is generally smaller than r_2. The chaotic local search is not performed every iteration in order to achieve a good compromise between the effectiveness of the search and the required computational effort; in fact, since the success of the local search is not guaranteed, it is more reasonable to make it happen randomly to save compuational time. Z is also a real number which is initialized randomly in the range $[0, 1]$ and it is updated with the following rule describing a chaotic logistic map (Jia et al., 2011)

$$Z_{t+1} = 4Z_t \cdot (1 - Z_t). \tag{24}$$

The value of Z is updated only if the chaotic local search succeds in finding a new global best, i.e., $J_{g1}^k < J_g^k$ (for a minimzation problem). In this case, the index t is increased by 1 and the new best CP ($\mathbf{X}_{g1}^k$) replaces the old one ($\mathbf{X}_g^k$).

2.4. Boundaries constraints handling

Boundaries violations can be managed in very different ways. For example, the variables exceeding the boundaries could be directly re-initialized or substituted by the lower or upper bound values. More complex strategies could be adopted, such as the harmony search-based algorithm (Kaveh & Talatahari, 2010b; Kaveh et al., 2013; Lee & Geem, 2004; Kaveh & Talatahari, 2009). In this case, when a variable exceeds the lower/upper bound defined by the user, it is either re-initialized or substituted by the corresponding value belonging to a CP chosen among the CM (or among its neighborhood). This strategy is ruled by two parameters: the CMCR, which is the Charged Memory Considering Rate, and the PAR, that is the Pitching Adjusting Rate. However, two further values should be decided. In order to make the algorithm automatically choose the best way to handle these violations, the following strategy is exploited,

$$\begin{cases} X_{i,m} = X_{CM_{r_i},m} & \text{if} \quad r_1^2 < r_2, \\ X_{i,m} = LB_m + r \cdot (UB_m - LB_m) & \text{otherwise.} \end{cases} \tag{25}$$

The index m is used to indicate the dimension associated to the outbound variable, while r, r_1 and r_2 are three real random numbers. Everytime a variable violates its corresponding boundary, a counter is updated ($c_{LB,m}$ in the case of a lower bound violation and $c_{UB,m}$ for an upper bound violation). At the end of an internal loop (i.e. before starting the successive external loop), these counters are used to make the algorithm understand if the boundaries could have been bad-defined by the user. If these counters exceed a maximum value C_{max}, the corresponding boundary is enlarged, as shown in Eq. (26).

$$\begin{cases} UB_m = UB_m \cdot 10 + \epsilon & \iff c_{UB,m} > C_{max} \ \wedge \ UB_m \geq 0 \\ LB_m = LB_m/10 - \epsilon & \iff c_{LB,m} > C_{max} \ \wedge \ LB_m \geq 0 \\ UB_m = UB_m/10 & \iff c_{UB,m} > C_{max} \ \wedge \ UB_m \leq 0 \\ LB_m = LB_m \cdot 10 & \iff c_{LB,m} > C_{max} \ \wedge \ LB_m \leq 0 \end{cases} \tag{26}$$

Note that the value of C_{max} is adaptive with respect to N_{CP} with the law

$$C_{max} = 0.1 \cdot k_f \cdot N_{CP}, \tag{27}$$

where k_f represents the index of the last internal iteration. This value could be different from G_k ($k_f \leq G_k$) because two exit strategies are considered, i.e., when the maximum iterations number (G_k) is reached or when a pre-defined tolerance is satisfied (refer to Sec 2.6). In Eq. (26), ϵ represents a small positive constant. This tool is very useful and could help the algorithm to find a global minimum when it is located on the boundaries or when it is not inside the search space (in these cases the CPs tend to move much more towards the boundaries and this could be an indication of a bad-defined boundary). Certainly, if the objective function has local minima inside the search space, it could be more difficult for the IMCSS to realize that the boundaries have been badly defined.

2.5. Entropy of the system

In order to have an idea of how the optimization is being performed by the IMCSS, a parameter, called here "entropy" (S), can be computed. The entropy

is defined as

$$S = \text{median}(\mathbf{J}^k) \tag{28}$$

Eq. (28) associates the entropy of the population to the median element of the objective functions vector. Obviously, if S decreases, it means that the CPs are closer to the best CP; thus, the algorithm is going to converge. Entropy of CPs can convey the idea of the distribution of the CPs with respect to the best one. It is important to note that, in order to state that the algorithm is performing well, the entropy needs to be decreasing. This is in contrast with the concept of entropy in Physics, which is always a non-decreasing function for natural transformations.

At each iteration, if S is non-decreasing (i.e., the disorder of CPs in not decreasing) a counter (c_S) is updated. At the end of an internal loop, the N_{CP} is increased if c_S overcomes a defined maximum value,

$$N_{CP}^{K+1} = \min[N_{CP}^{K} + r_{in}[W;\ 3\lceil\log(D+1)\rceil];\ \lceil 1.1 N_{CP,max}\rceil] \ \text{ if } \ c_S/k_f > 0.5. \tag{29}$$

In Eq. (29), r_i represents a random integer number between the two in brackets. The proposed entropy-based strategy represents an opportunity to improve the results when the behaviour of the algorithm is not completely satisfactory according to the criterion of Eq. (29). Indeed, even though it does not guarantee to find the optimal solution of a generic problem, a greater number of CPs can improve the convergence of the next loops of the algorithm with respect to the previous ones.

2.6. End of the algorithm

As already mentioned, the internal loop ends when the maximum iterations number is reached or when a pre-defined tolerance (δ_{user}) is achieved. The same occurs with the external loop. The tolerances can be defined in many different way by taking into account the objective functions of the CPs, such

Algorithm 1 IMCSS algorithm

Preliminary parameters initialization: N_{CP}, G_k, $G_{k,f}$, G_K, $k_{v,0}$, $k_{v,f}$, $k_{a,0}$, $k_{a,f}$, k_{ar}, Z (Eqs. (2)-(5), (17), (22));
while ($K \leq G_K \ \wedge \ \delta_K \geq \delta_{user}$) **do**
    CPs position and velocities initialization. Objective functions computation;
    If $K > 1$: Substitute the first CP with the best CP of the previous internal loop;
    Initialize the Charged Memory (CM) and compute the entropy by means of Eq (28);
    while ($k \leq G_k \ \wedge \ \delta_k \geq \delta_{user}$) **do**
        Update acceleration and velocity coefficients (Eqs. (20) and (21));
        Compute the total force, the new positions and velocities (Eqs. (15),(18) and (19));
        Handle boundaries violations (Eq. (25)) and update the counters ($\mathbf{c}_{UB}$ and $\mathbf{c}_{LB}$);
        Objective functions computation and CLS (Eq (23)). Update CM;
        Calculate the internal loop tolerance (δ_k) and update the entropy S and c_S;
        k=k+1;
    end while
    Store the global best CP of the current internal loop;
    Check the boundaries enlargement and non-decreasing entropy (Eq (29)) conditions;
    Calculate the external loop tolerance (δ_K) using $\mathbf{J}_g^K$;
    K=K+1;
end while
Retrieve the global best CP.

as by computing the absolute error, the relative error, the standard deviation among a set of CPs (N_δ), etc.. Throughout this paper, the tolerance (δ) must be lower than 10^{-10} and it is defined as the standard deviation of the last N_δ objective functions, i.e.,

$$\delta = \sqrt{\frac{\sum_{k=1}^{N_\delta}(J_g^k - J_g^{mean})^2}{N_\delta}}, \tag{30}$$

where J_g^{mean} is the mean value of the considered N_δ objective functions. Furthermore, if $N_{CP} < N_{CP,max}$, the last external loop is performed with an increased number of CPs, which is updated with the same law as the one in Eq. (29), in order to ensure a better convergence. The pseudocode and the flow chart of IMCSS are provided respectively in Algorithm 1 and Fig.1 to summarize the whole architecture of the proposed approach.

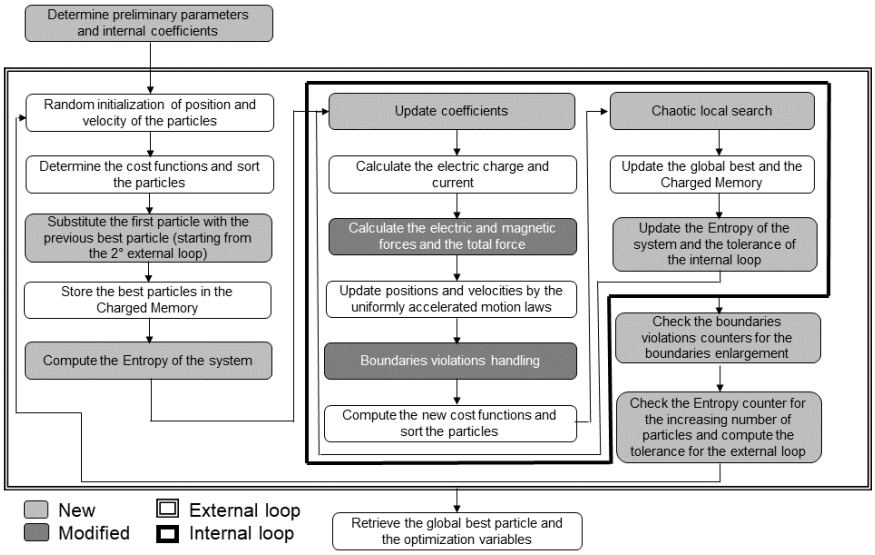

Figure 1: IMCSS flow chart with novelties and modifications wrt to the original MCSS.

3. IMCSS performance analysis

In order to test the effectiveness of the proposed IMCSS algorithm, a first test is performed on some benchmark functions, which are optimization functions whose global minima are well-known. This analysis allows to test the efficiency of IMCSS in finding the global minima. Table 1 shows the considered benchmark functions (some of them are the same as in the original paper about MCSS (Kaveh et al., 2013) and other ones have been added). Many different functions are considered, such as unconstrained and constrained, continous and non-separable, convex and non-convex, multimodal and unimodal, differentiable and non-differentiable, in order to take into account many heterogeneous conditions (a list of the benchmark functions and their features can be found freely online [1]).

In particular, their analytical expressions, the dimension of the problems, the lower and upper bounds of the variables and the global minima are indicated in

[1] http://benchmarkfcns.xyz/fcns, accessed on April 26, 2019

Table 1: List of the considered benchmark functions.

Function	Name	Expression	D	Range	F^*_{min}
BF1	Aluffi*i*Pentiny	$f(\mathbf{X}) = \frac{1}{4}x_1^4 - \frac{1}{2}x_1^2 + \frac{1}{10}x_1 + \frac{1}{2}x_2^2$	2	$[-10, 10]^D$	-0.352386
BF2	Bohachevsky 1	$f(\mathbf{X}) = x_1^2 + 2x_2^2 - \frac{3}{10}\cos(3\pi x_1) - \frac{4}{10}\cos(4\pi x_1) + \frac{7}{10}$	2	$[-100, 100]^D$	0
BF3	Bohachevsky 2	$f(\mathbf{X}) = x_1^2 + 2x_2^2 - \frac{3}{10}\cos(3\pi x_1) \cdot \cos(4\pi x_1) + \frac{3}{10}$	2	$[-100, 100]^D$	0
BF4	Becker and Lago	$f(\mathbf{X}) = (\lvert x_1\rvert - 5)^2 + (\lvert x_2\rvert - 5)^2$	2	$[-10, 10]^D$	0
BF5	Branin	$f(\mathbf{X}) = \left(x_2 - \frac{5.1}{4\pi^2}x_1^2 + \frac{5}{\pi}x_1 - 6\right) + 10 \cdot \left(1 - \frac{1}{8\pi}\right)\cos(x_1) + 10$	2	$x_1 \in [-5, 10]$ $x_2 \in [0, 15]$	0.397887
BF6	Camel	$f(\mathbf{X}) = 4x_1^2 - 2.1x_1^4 + \frac{1}{3}x_1^6 + x_1x_2 - 4x_2^2 + 4x_2^4$	2	$[-5, 5]^D$	-1.031628
BF7	Cb3	$f(\mathbf{X}) = 2x_1^2 - 1.05x_1^4 + \frac{1}{6}x_1^6 + x_1x_2 + x_2^2$	2	$[-5, 5]^D$	0
BF8	Cosine mixture	$f(\mathbf{X}) = \sum_{m=1}^{D}(x_m^2 - \frac{1}{10}\cos(5\pi x_m))$	4	$[-1, 1]^D$	-0.4
BF9	DeJoung	$f(\mathbf{X}) = x_1^2 + x_2^2 + x_3^2$	3	$[-5.12, 5.12]^D$	0
BF10	Exponential	$f(\mathbf{X}) = -\exp(-0.5\sum_{m=1}^{D} x_m^2)$	4	$[-1, 1]^D$	-1.0
BF11	Goldstein and price	$f(\mathbf{X}) = [1 + (x_1 + x_2 + 1)^2(19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1x_2 + 3x_2^2)] \cdot [30 + (2x_1 - 3x_2)^2(18 - 32x_1 + 12x_1 + 48x_2 - 36x_1x_2 + 27x_2^2)]$	2	$[-2\ ,\ 2]^D$	3
BF12	Grienwank	$f(\mathbf{X}) = 1 + \frac{1}{200}\sum_{m=1}^{D} x_m^2 - \prod_{m=1}^{D}\cos\left(\frac{x_m}{\sqrt{d}}\right)$	10	$[-600, 600]^D$	0
BF13	Hartman3	$f(\mathbf{X}) = -\sum_{s=1}^{4} c_s \exp[-\sum_{m=1}^{D}(a_{sd}(x_m - p_{sd})^2)]$, $a = \begin{bmatrix} 3 & 10 & 30 \\ 0.1 & 10 & 35 \\ 3 & 10 & 30 \\ 0.1 & 10 & 35 \end{bmatrix}$, $c = \begin{bmatrix} 1 \\ 1.2 \\ 3 \\ 3.2 \end{bmatrix}$, p= $\begin{bmatrix} 0.3689 & 0.117 & 0.2673 \\ 0.4699 & 0.4387 & 0.747 \\ 0.1091 & 0.8732 & 0.5547 \\ 0.03815 & 0.5743 & 0.8828 \end{bmatrix}$	3	$[-10, 10]^D$	-3.862782

Function	Name	Expression	D	Range	F^*_{min}
BF14	Hartman6	$f(\mathbf{X}) = -\sum_{s=1}^{4} c_s \exp[-\sum_{m=1}^{D}(a_{sd}(x_m - p_{sd})^2)]$, $a = \begin{bmatrix} 10 & 3 & 17 & 3.5 & 1.7 & 8 \\ 0.05 & 10 & 17 & 0.1 & 8 & 14 \\ 3 & 3.5 & 1.7 & 10 & 17 & 8 \\ 17 & 8 & 0.05 & 10 & 0.1 & 14 \end{bmatrix}$ $c = \begin{bmatrix} 1 \\ 1.2 \\ 3 \\ 3.2 \end{bmatrix}$, $p = \begin{bmatrix} 0.1312 & 0.1696 & 0.5569 & 0.0124 & 0.8283 & 0.5886 \\ 0.2329 & 0.4135 & 0.8307 & 0.3736 & 0.1004 & 0.9991 \\ 0.2348 & 0.1451 & 0.3522 & 0.2883 & 0.3047 & 0.6650 \\ 0.4047 & 0.8828 & 0.8732 & 0.5743 & 0.1091 & 0.0381 \end{bmatrix}$	6	$[0,1]^D$	-3.322368
BF15	Rosenbrock	$f(\mathbf{X}) = \sum_{m=1}^{D-1}[100 \cdot (x_{d+1} - x_m^2)^2 + (x_m - 1)^2]$	2	$[-30,30]^D$	0
BF16	Rastrigin	$f(\mathbf{X}) = 10D + \sum_{m=1}^{D}(x_m^2 - 10 \cdot \cos(2\pi x_m))$	3	$[-5.12,5.12]^D$	0
BF17	Mishra	$f(\mathbf{X}) = \sin(x_2)\exp[(1-\cos x_1)^2] + \cos(x_1)\exp[(1-\sin(x_2)^2)]$ $+(x_1 - x_2)^2$, subject to: $(x_1+5)^2 + (x_2+5)^2 < 25$	2	$x_1 \in [-10,50]$ $x_2 \in [-6.5,50]$	-106.764537
BF18	Beale	$f(\mathbf{X}) = (1.5 - x_1 + x_1x_2)^2 + (2.25 - x_1 + x_1x_2^2)^2 + (2.625 - x_1 + x_1x_2^3)^2$	2	$[-40,40]^D$	0
BF19	Townsend	$f(\mathbf{X}) = -\cos[(x_1 - 0.1)x_2]^2 - x_1\sin(3x_1 + x_2)$ subject to: $x_1^2 + x_2^2 - (2\sin(t))^2 -$ $[2\cos(t) - \frac{1}{2}\cos(2t) - \frac{1}{4}\cos(3t) - \frac{1}{8}\cos(4t)]^2 < 0$ where $t = \text{atan2}(x_1, x_2)$	2	$x_1 \in [-2.25,2.5]$ $x_2 \in [-2.5,1.75]$	-2.023988
BF20	Ackley	$f(\mathbf{X}) = -20\exp[-0.2\sqrt{\frac{1}{D}\sum_{m=1}^{D}x_m^2}] - \exp[\frac{1}{D}\sum_{m=1}^{D}\cos(2\pi x_m)] + e + 20$	10	$[-32,32]^D$	0
BF21	Cross-in-tray	$f(\mathbf{X}) = -0.0001\left[\left\lvert\sin(x_1)\sin(x_2)\exp\left(\left\lvert 100 - \frac{\sqrt{x_1^2+x_2^2}}{\pi}\right\rvert\right)\right\rvert + 1\right]^{0.1}$	2	$[-10,10]^D$	-2.062612
BF22	Eggholder	$f(\mathbf{X}) = -(x_2 + 47)\sin\sqrt{\lvert\frac{x_1}{2} + x_2 + 47\rvert} - x_1\sin\sqrt{\lvert x_1 - x_2 - 47\rvert}$	2	$[-512,512]^D$	-959.6407
BF23	Holder table	$f(\mathbf{X}) = -\left\lvert\sin(x_1)\cos(x_2)\exp\left(\left\lvert 1 - \frac{\sqrt{x_1^2+x_2^2}}{\pi}\right\rvert\right)\right\rvert$	2	$[-10,10]^D$	-19.208503
BF24	Levi13	$f(\mathbf{X}) = \sin^2(3\pi x_1) + (x_1-1)^2[1+\sin^2(3\pi x_2)] + (x_2-1)^2[1+\sin^2(3\pi x_2)]$	2	$[-10,10]^D$	0

Table 2: Statistical results obtained by IMCSS through 1000 independent runs on the benchmark functions.

Fun.	Best	Worst	Mean	Std	$\left\|F^*_{min} - Best\right\|$	MCSS
BF1	-0.352386	-0.352386	-0.352386	5.160994E-10	0	-0.34974
BF2	0	1.268790E-07	9.910206E-09	1.749269E-08	0	1.435E-06
BF3	0	1.569533E-07	9.253131E-09	1.656263E-08	0	5.358E-06
BF4	0	7.068693E-10	1.085573E-12	2.306068E-11	0	2.542E-08
BF5	0.397887	0.397887	0.397887	1.656622E-09	0	0.401754
BF6	-1.031628	-1.031628	-1.031628	4.095771E-09	0	-1.031598
BF7	0	8.843861E-09	1.721074E-10	6.158793E-10	0	2.427E-08
BF8	-0.4	-0.4	-0.4	5.664971E-15	0	-0.398751
BF9	0	1.837917E-07	1.341720E-08	2.219142E-08	0	4.731E-06
BF10	-1	-1	-1	0	0	-0.999971
BF11	3	3	3	8.731835E-13	0	3.155378
BF12	1.946010E-10	0.145619	1.460866E-04	0.004605	1.946010E-10	6.815E-08
BF13	-3.862782	-3.862780	-3.862782	1.383581E-07	0	-3.861415
BF14	-3.321995	-1.267677	-3.275728	0.123722	3.728284E-04	-3.320421
BF15	0	2.553494E-08	1.094324E-09	2.628318E-09	0	0.074523
BF16	0	0.401014	0.002139	0.016957	0	-
BF17	-106.764537	-106.764535	-106.764537	1.248452E-07	0	-
BF18	4.309666E-15	7.867842E-09	3.084089E-10	6.750814E-10	4.309666E-15	-
BF19	-2.023988	-1.659511	-2.002659	0.062348	0	-
BF20	9.523413E-10	1.155187	0.005338	0.063119	9.523413E-10	-
BF21	-2.062612	-2.062612	-2.062612	1.494329E-10	0	-
BF22	-959.640662	-894.578900	-957.825537	8.712252	3.727915E-05	-
BF23	-19.208503	-19.208503	-19.208502	6.413715E-09	0	-
BF24	0	5.496055E-08	1.643349E-09	4.384696E-09	0	-

Table 1. Table 2 reports some statistical results (the best, the worst, the mean values, the standard deviation and the absolute error with respect to the real minimum F^*_{min}) obtained through 1000 independent runs on the benchmark functions, without excluding the outliers. In Table 2, the error between the computed and the real minimum is 0 considering the limits on the decimal digits that have been found online. The IMCSS ends when the maximum iterations number is reached or when the condition shown in Eq. (30) is fulfilled. For this analysis, N_δ is set equal to 3 for both the internal and external loops. The last column of Table 2 also shows the results of the optimization on some Benchmark functions (BF1-15) with the original algorithm MCSS. Considering that the results of the optimization in (Kaveh et al., 2013) are computed with 10 particles,a maximum number of iterations equal to 100 and without taking into account a statistics on multiple runs, the results between MCSS and the proposed IMCSS can only be compared by looking at the mean values of the statistics. As can be noticed comparing the MCSS results with the mean values obtained with IMCSS, the proposed algorithm generally outperforms the MCSS, except in BF14 (Hartman6) and BF12 (Grienwank). However, for BF12, the comparison cannot precisely be done, since in the original paper of MCSS the dimension associated to this problem is only 2 compared to dimension 10 of the present paper. For the common Benchmark functions between MCSS and IMCSS, two different Wilcoxon tests are carried out by considering the mean values and then the best values of IMCSS. For the first case, assuming an α value equal to 0.05 and that the null hypothesis corresponds to a not significantly different behaviour of the two algorithms, the obtained p value is 0.02557 which is smaller than 0.05. Therefore, the null hypothesis is rejected and it can be concluded that the two algorithms behave differently. In particular, IMCSS outperforms MCSS since its mean value is lower. The same considerations can be done considering the best values of IMCSS for the comparison: even in this case, the p value ($6.1035 \cdot 10^{-5}$) is lower than the α value (0.05). For what concerns the other Benchmark functions (BF16-24), they have been added to the analysis with the only purpose to test the algorithm with other functions

with different features, so they can be considered for future comparisons.

A particular test for the Beale function (BF18, $F_{min} = 0, x_{1,min} = 3, x_{2,min} = 0.5$) is conducted to show the effectiveness of the boundaries handling strategy introduced in Sec. 2.4. Let the lower and upper bounds of the variables be bad defined: $\mathbf{LB} = [0, -0.03]$ and $\mathbf{UB} = [2, 0.03]$. As one can note, the correct variables, which correspond to the global minimum of the function, are not within the defined boundaries. If no strategy is performed and the boundaries remain fixed, the computed global minimum is 0.64769574. Instead, if the strategy about the boundaries handling explained in Sec. 2.4 is applied, the IMCSS is able to autonomously understand that the global minimum cannot be obtained with the defined boundaries. In particular, what happens is that most of the variables tend to move and accumulate towards the lower and upper bounds values, suggesting that they want to move outside the boundaries to search for the correct global minimum. Table 3 shows the time history of the optimization problem for the Beale function with bad-defined boundaries: $\mathbf{c}_{UB}$ and $\mathbf{c}_{LB}$ indicate the counters for the violation of the lower and upper bounds of the variables, whereas the last two columns show the changes of the bounds. As can be seen in the first row of the table, both the components of the violated upper bound counter $\mathbf{c}_{UB}$ are greater than the maximum admitted value C_{max}

Table 3: Results of the boundary constraints handling technique for the Beale function with $N_{CP} = 20$.

G_K	$G_{k,f}$	J_g	δ	C_{max}	$\mathbf{c}_{UB}$	$\mathbf{c}_{LB}$	**UB**	**LB**
1	128	0.64769574	8.78738E-11	**256**	**[324,267]**	[25,33]	[2,0.03]	[0,-0.03]
2	209	0.10013305	4.23267E-11	**418**	[9,**523**]	[11,42]	[**20**,**0.3**]	[0,-0.03]
3	216	0.00000000	4.45473E-11	432	[12,12]	[5,10]	[20,**3**]	[0,-0.03]
4	226	0.00000000	7.10376E-11	452	[20,10]	[12,9]	[20,3]	[0,-0.03]
5	228	0.00000000	5.78289E-11	456	[13,8]	[14,8]	[20,3]	[0,-0.03]
6	259	0.00000000	9.29849E-11	518	[8,8]	[4,10]	[20,3]	[0,-0.03]
7	185	0.00000000	3.31962E-11	407	[12,10]	[4,8]	[20,3]	[0,-0.03]

after the first external loop. This causes the enlargement of the two components of the upper bound, which becomes $[20, 0.3]$. This new upper bound includes $x_{1,min}$, but not $x_{2,min}$. In the second external loop, the value of C_{max} is updated according to Eq. (27). Since $c_{UB,2} = 523 > C_{max} = 418$, the second component of **UB** is again modified and $\mathbf{c}_{UB}$ becomes $[20, 3]$. Finally, this is a correct boundary that includes both the $x_{1,min}$ and $x_{2,min}$. As illustrated in the table, the third external loop already succeeds in finding the correct global minimum. This example has shown the ability of IMCSS to find the correct global with bad defined initial boundaries. Hence, IMCSS can overcome problems coming from bad-defined boundaries due to a not perfect knowledge of the optimization problem.

For what concerns the efficiency of the CLS technique, another analysis is performed on the 24 Benchmark functions considering 1000 independent runs for each function. Two Montecarlo analysis are performed: the first one is carried out without using the CLS, whereas the second employs the CLS. The same random numbers for the two Montecarlo are generated in order to actually test the reliability of the test. The results show that, for 17 functions out of 24, the CLS helps for the majority of the runs (effective runs ≥ 500) to improve the convergence to the global optimum, which is considered to be obtained when the algorithm reaches the global best with an error of 10^{-4} with respect to the real global minimum. The functions for which the CLS does not provide a significant real improvement (effective runs < 500) are: BF12, 14, 15, 16, 19, 20 and 22. Three examples about the comparison of the performances of IMCSS with and without CLS, regarding the functions BF7, BF13, BF17, are shown in Fig. 2. As can be noted, for all the cases the chaotic local search allows to reach the convergence much faster; in particular, for BF17 the advantage of exploiting the CLS inside the algorithm is evident because the global optimum is approached since the beginning of the optimization procedure. This analysis shows that the CLS can actually speed up the convergence for most of the cases; thus, it is convenient to employ it at the expense of a greater number of functions evaluations per iteration.

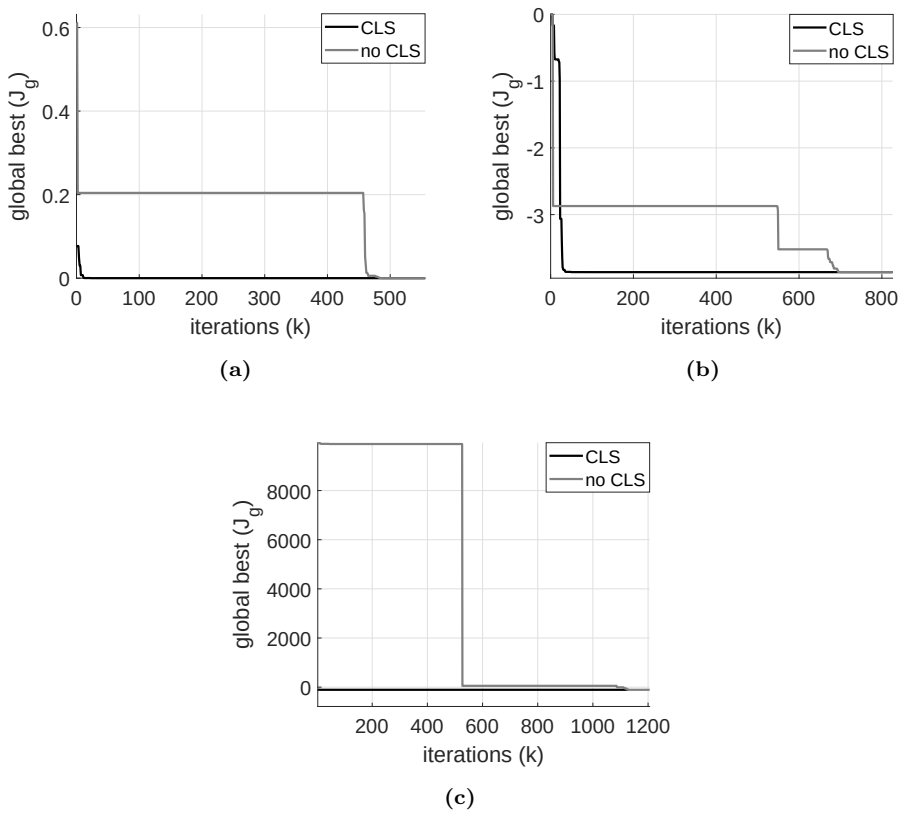

Figure 2: Convergence of the algorithm with and without the Chaotic Local Search for BF7 (a), BF13 (b) and BF17 (c).

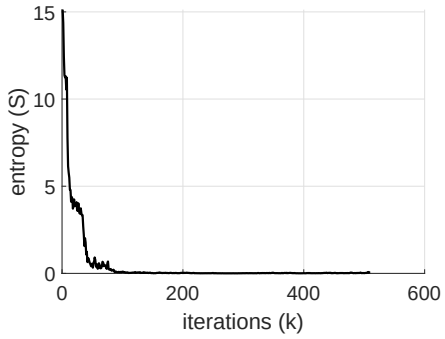

Figure 3: Entropy of the system as a function of the iteration number (k) for BF20.

Another important point is related to the increase of the N_{CP} because of the non-decreasing entropy of the system. For example, considering the Ackley function (BF20) after the second external loop , with $N_{CP} = 30$, it can happen that the entropy is non-decreasing for more than 50% of the total iterations, as reported in Fig. 3 (256 iterations over a total of 510 had a non-decreasing entropy). Non-decreasing entropy could mean that the algorithm has already found a possible candidate for the global minimum, thus the entropy does not decrease anymore, but it could also mean that it has stalled in a wrong minimum. Therefore, in order to increase the probability to converge to the real global minimum, according to what has been said in the previous Section, the number of CPs is increased, decreasing the failure possibilities of the algorithm. It is important to highlight that this procedure can help the IMCSS to speed up the convergence, but it could also be not determinant (it is only assumed that a better and faster convergence can be achieved with a greater number of CPs). These above-mentioned examples shows the effectiveness and abilities of the proposed algorithm to try to overcome some difficulties in finding the optimum.

While the previous tests on the Benchmark functions aim at showing the potentialities of the proposed algorithm and its comparison with the original MCSS, a second analysis is carried out on well-known engineering optimization problems, such as the tension/compression spring design, the welded beam design and the pressure vessel design problems. This analysis allows to compare IMCSS to other works in literature which employ very different methods of optimization, from mathematical-based to metaheuristic and evolutionary algorithms. For the sake of conciseness, the descriptions of these three problems are not reported here; for further details on the mathematical transcriptions of these problems and the related boundaries of the optimization variables the reader can refer to (Kaveh et al., 2013). From Table 4, which represents the statistics for the tension/compression spring design problem, it is possible to see that IMCSS outperforms all the algorithms apart from the last three. However, for these three cases, their mean values are always greater than the mean and also the worst values found by IMCSS, thus proving a more reliability of

the proposed algorithm although the best value of IMCSS is less than the other three. The statistics for the welded beam design problem is represented in Table 5, which shows that the best value of IMCSS is better than those of the other works, but the mean value is higher than the last three approaches. By looking at the mean value, since it is closer to the best value than the worst one, it is possible to understand that there are only few runs for which the minima objective functions reach high values. Finally, the statistics on the pressure vessel design problem, illustrated in Table 6, shows that IMCSS outperforms all the other algorithms with a significant lower value of the best function. As can be deduced from this analysis, IMCSS can outperform (or at least be comparable to) many algorithms, thus proving its reliability and competitiveness with other optimization approaches while adding at the same time the important feature of a more user-friendly usage.

Table 4: Comparison between IMCSS and other algorithms on the tension/compression spring design problem.

Methods	Best	Mean	Worst	Std
(Belegundu, 1983)	0.0128334	N/A	N/A	N/A
(Arora, 1989)	0.0127303	N/A	N/A	N/A
(Coello, 2000)	0.0127048	0.012769	0.012822	3.9390E-05
(Coello & Montes, 2002)	0.0126810	0.0127420	0.012973	5.9000E-05
(He & Wang, 2007)	0.0126747	0.012730	0.012924	5.1985E-05
(Mezura-Montes & Coello, 2008)	0.012698	0.013461	0.16485	9.6600E-04
(Kaveh & Talatahari, 2010a)	0.0126432	0.012720	0.012884	3.4888E-05
(Kaveh & Talatahari, 2010b)	0.0126384	0.012852	0.013626	8.3564E-05
(Kaveh et al., 2013)	0.0126069	0.012712	0.012982	4.7831E-05
Present work	0.0126652	0.012665	0.012666	5.5318E-08

4. Optimal Control Problem Statement

The description of the optimal control problem related to satellite formation reconfiguration is provided in this section. Minimum time manoeuvres will be addressed for a formation of two satellites, the chief c and the deputy d. The

Table 5: Comparison between IMCSS and other algorithms on the welded beam design problem.

Methods	Best	Mean	Worst	Std
(Ragsdell & Phillips, 1976)	2.3815	N/A	N/A	N/A
(Deb, 1991)	2.433116	N/A	N/A	N/A
(Coello, 2000)	1.748309	1.771973	1.785835	0.011220
(Coello & Montes, 2002)	1.728226	1.792654	1.993408	0.074713
(He & Wang, 2007)	1.728024	1.748831	1.782143	0.012926
(Mezura-Montes & Coello, 2008)	1.737300	1.813290	1.994651	0.070500
(Kaveh & Talatahari, 2010a)	1.724918	1.729752	1.775961	0.009200
(Kaveh & Talatahari, 2010b)	1.724866	1.739654	1.759479	0.008064
(Kaveh et al., 2013)	1.724855	1.735374	1.750127	0.007571
Present work	1.724852	1.762985	2.640950	0.156215

Table 6: Comparison between IMCSS and other algorithms on the pressure vessel design problem.

Methods	Best	Mean	Worst	Std
(Sandgren, 1988)	8129.1036	N/A	N/A	N/A
(Kannan & Kramer, 1994)	7198.0428	N/A	N/A	N/A
(Deb, 1997)	6410.3811	N/A	N/A	N/A
(Coello, 2000)	6288.7445	6293.8432	6308.1497	7.4133
(Coello & Montes, 2002)	6059.9463	6177.2533	6469.3220	130.9297
(He & Wang, 2007)	6061.0777	6147.1332	6363.8041	86.4545
(Mezura-Montes & Coello, 2008)	6059.7456	6850.0049	7332.8798	426.0000
(Kaveh & Talatahari, 2010a)	6059.7258	6081.7812	6150.1289	67.2418
(Kaveh & Talatahari, 2010b)	6059.0888	6067.9062	6085.4765	10.2564
(Kaveh et al., 2013)	6058.9710	6063.1798	6074.7391	9.73494
Present work	5885.3327	5885.4119	5886.7128	0.09339

equations of motion of the system, referred to an Earth-centered inertial (ECI) coordinate system $\mathscr{I} = \{X_{\mathscr{I}}, Y_{\mathscr{I}}, Z_{\mathscr{I}}\}$ shown in Fig. 4, are

$$\ddot{\mathbf{r}} = -\frac{\mu}{r^3}\mathbf{r} + \mathbf{u} + \mathbf{f} \tag{31}$$

where $\mu = 3.986 \cdot 10^5$ km^3/s^2 is the Earth gravitational parameter, $\mathbf{r}$ is the

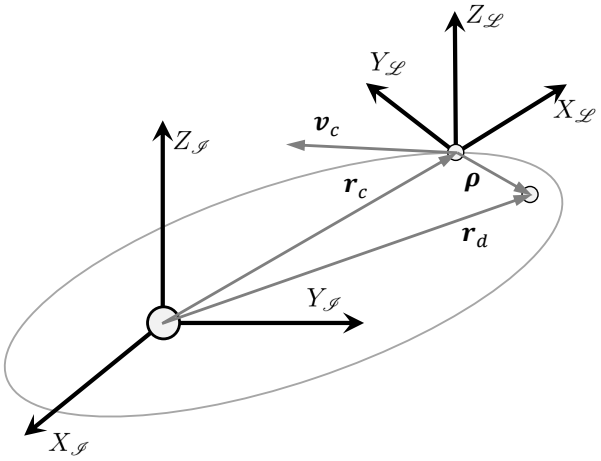

Figure 4: ECI coordinate system $\mathscr{I}$ and LVLH coordinate system $\mathscr{L}$.

position vector of the satellite (r is its magnitude), $\mathbf{u}$ is the external acceleration (provided by the engines), and $\mathbf{f}$ is the perturbation acceleration (depending on the environmental forces). Let $\boldsymbol{\rho} = \mathbf{r}_d - \mathbf{r}_c = [\rho_x, \rho_y, \rho_z]^T$ be the relative distance between the chief and the deputy. The relative motion is usually described in the local-vertical/local-horizontal (LVLH) reference frame $\mathscr{L} = \{X_{\mathscr{L}}, Y_{\mathscr{L}}, Z_{\mathscr{L}}\}$, shown in Fig 4, with $X_{\mathscr{L}}$ pointing from the center of the Earth to the origin of $\mathscr{L}$, $Z_{\mathscr{L}}$ perpendicular to the orbital plane and $Y_{\mathscr{L}}$ completing the right-hand Cartesian coordinate system. This reference system is centered on the chief satellite. Consider the chief satellite on a circular orbit (which implies $\omega = \sqrt{\mu/r^3}$) with inclination i_c and let the J_2 gravitational effect in $\mathbf{f}$ be considered in the following mathematical model (see (David & McClain, 2007) for further astrodynamical details). Imposing $\mathbf{u}_c = 0$, the relative motion is described by the SS model developed by Schweighart and Sedwick (Schweighart & Sedwick, 2001, 2002),

$$\begin{cases} \ddot{\rho}_x - 2\bar{m}\dot{\rho}_y - (4\bar{m}^2 - \bar{n}^2)\rho_x = u_x \\ \ddot{\rho}_y + 2\bar{m}\dot{\rho}_x = u_y \\ \ddot{\rho}_z + (2\bar{m}^2 - \bar{n}^2)\rho_z = u_z \end{cases} \tag{32}$$

where $\bar{m} = \omega\sqrt{1+k_{J_2}}$, $\bar{n} = \omega\sqrt{1-k_{J_2}}$, $k_{J_2} = (3J_2R_e^2/8r_c^2)[1+3\cos(2i_c)]$ and R_e is the Earth radius. Note that the in-plane motion along $X_{\mathscr{L}}$ and $Y_{\mathscr{L}}$ is decoupled from the out-of-plane motion along $Z_{\mathscr{L}}$. It easy to see that Eq. (32) can be also written in the state-space form in the following form

$$\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{u} \tag{33}$$

where matrices $\mathbf{A}$ and $\mathbf{B}$ are computed from Eq. (32), $\mathbf{x} = [\boldsymbol{\rho}_x^{\mathsf{T}}, \dot{\boldsymbol{\rho}}_x^{\mathsf{T}}]^{\mathsf{T}}$ and $\mathbf{u} = [u_x, u_y, u_z]^{\mathsf{T}}$. Let t_0 and t_f be the initial and the desired final relative state; the time-optimal problem considered in this paper is defined as follows

$$\begin{cases} \text{Find } (\mathbf{x}, \mathbf{u}, t_f)^* \\ \text{minimizing } J = t_f - t_0 \\ \text{subject to, } \forall t \in [t_0, t_f], \\ \text{dynamical constraints : } \dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{u} \\ \text{boundary conditions : } \mathbf{x}(t_0) = \mathbf{x}_0,\ \mathbf{x}(t_f) = \mathbf{x}_f, \\ \text{control constraint : } ||\mathbf{u}(t)||_\infty - u_{max} \leq 0 \end{cases} \tag{34}$$

where the infinity norm is defined as $\|\boldsymbol{u}(t)\|_\infty = \max\{|u_x(t)|, |u_y(t)|, |u_z(t)|\}$. The reconfiguration manoeuvre must be accomplished in the shortest final time t_f^*.

4.1. Inverse dynamics technique

The direct dynamics is usually employed for most of the applications; it consists on obtaining the trajectory via an integration process under an applied control. Instead, the inverse dynamics technique is exploited in this paper since the external control $\mathbf{u}(t)$ can be written as a function of the state. In this case, the state vector $\mathbf{x}$ is approximated by some functions or curves. The optimization algorithm searches for the optimal values of the state approximation coefficients describing the curve, or the function, of each state component. The control is obtained from the state and its time derivatives (the required order of derivatives depends on the problem). The inverse dynamics approach overcomes some direct-dynamics issues such as low computational speeds and integration

of numerical errors and has a reduced number of optimization parameters with respect to traditional methods (collocation and pseudospectral). All these features allow to have a very good compromise between the computational time and the optimality of the results. It is important to note that, because of the approximation of the position vector (and its first and second derivatives) the inverse dynamics technique usually provides a sub-optimal solution representing a good compromise between reliability and complexity. Within the frame of the inverse dynamics technique, a particular subclass is the differential flatness (DF) formulation (Fliess et al., 1995), which is based on the so-called flat output $\mathbf{y}(t)$ (with the same dimension of the control $\mathbf{u}(t)$). If such a flat output exists, then the state and the control can be written as functions of $\mathbf{y}$,

$$\mathbf{x} = \mathbf{a}(\mathbf{y}, \dot{\mathbf{y}}, ..., \mathbf{y}^{(\beta)}), \quad \mathbf{u} = \mathbf{b}(\mathbf{y}, \dot{\mathbf{y}}, ..., \mathbf{y}^{(\beta+1)}) \tag{35}$$

For our optimization problem, let $\mathbf{y} = \boldsymbol{\rho}$. According to this choice, $\beta = 1$ and the full state $\mathbf{x}$ is obtained via time differentiation of $\boldsymbol{\rho}$, whereas $\mathbf{u} = \mathbf{b}(\boldsymbol{\rho}, \dot{\boldsymbol{\rho}}, \ddot{\boldsymbol{\rho}})$. Therefore, the optimization problem, illustrated in Eq. (34), can be rewritten as a function of the flat output as

$$\begin{cases} \text{Find } (\boldsymbol{\rho}, t_f)^* \\ \text{minimizing } J = t_f - t_0 \\ \text{subject to, } \forall t \in [t_0, t_f], \\ \text{boundary conditions : } \boldsymbol{\rho}(t_0) = \boldsymbol{\rho}_0,\ \boldsymbol{\rho}(t_f) = \boldsymbol{\rho}_f, \\ \qquad\qquad\qquad\qquad \dot{\boldsymbol{\rho}}(t_0) = \dot{\boldsymbol{\rho}}_0,\ \dot{\boldsymbol{\rho}}(t_f) = \dot{\boldsymbol{\rho}}_f, \\ \text{control constraint : } \left|\left|\mathbf{b}(\boldsymbol{\rho}, \dot{\boldsymbol{\rho}}, \ddot{\boldsymbol{\rho}})\right|\right|_\infty - u_{max} \leq 0 \end{cases} \tag{36}$$

As it can be seen, the dynamics constraint does no longer appear in the new formulation of the optimization problem, because it is satisfied a priori. Note that Eq. (36) holds for the real (but unknown) flat output $\boldsymbol{\rho}(t)$ and for its numerical approximation $\boldsymbol{\rho}_N(t)$. In order to make the problem as simple as possible, the approximation curves can be chosen to directly fulfill also the boundary conditions, which are linked to a clever choice of the approximating polynomials

coefficients. Thus, calling with D_B the set of approximating function satisfying the boundary constraints, the problem is reformulated as follows

$$\begin{cases} \text{Find } (\boldsymbol{\rho}_N, t_f)^* \text{ such that } \boldsymbol{\rho}_N \in D_B \\ \text{minimizing } J = t_f - t_0 \\ \text{subject to, } \forall t \in [t_0, t_f], \\ \text{control constraint : } \left|\left|\mathbf{b}(\boldsymbol{\rho}_N, \dot{\boldsymbol{\rho}}_N, \ddot{\boldsymbol{\rho}}_N)\right|\right|_\infty - u_{max} \leq 0 \end{cases} \tag{37}$$

By employing the DF formulation, three main advantages can be highlighted:

1. The minimum number of optimization parameters is employed.
2. The control policy is obtained in an analytical closed form avoiding the integration of dynamics equations (thus saving also computational time).
3. Initial and final conditions are automatically respected since they are imposed a priori in the polynomial approximation of $\boldsymbol{\rho}$.

4.2. B-spline curves approximation

Many interpolating functions can be chosen to approximate the trajectory of the deputy satellite: this choice can strongly affect the results of the optimization problem. Among the most used approximation, there are the Chebyshev polynomials and the B-spline curves. In this paper, B-spline curves are employed, instead of Chebyshev polynomials, because it has already been demonstrated that B-spline curves perform better and allow to obtain more accurate results in this kind of optimization problem (Parente et al., 2018). The B-spline $\mathcal{B}(\lambda; \boldsymbol{a}, \mathbf{K})$ is characterised by a strictly increasing independent variable $0 \leq \lambda \leq 1$, the coefficient vector $\boldsymbol{a} = [a_0, a_1, ..., a_{N_P-1}]$ and the Knot vector $\mathbf{K}$, made up of $m+1$ components,

$$\mathbf{K} = \{k_n, n = 0, ..., m | k_0 = 0, k_m = 1, k_n \leq k_{n+1}\} \tag{38}$$

Here, N_P is the number of interpolating parameters and $m = N_P + \mathcal{D}$, where $\mathcal{D}$ is the degree of the polynomials. Hence, the l-th component of the approximated

relative displacement $\rho_{N;l}$ can be expressed as

$$\rho_{N;l}(\lambda) = \mathcal{B}(\lambda; \boldsymbol{a}, \mathbf{K}) = \sum_{n=0}^{N_P-1} a_n \mathcal{N}_{n,\mathcal{D}}(\lambda; \mathbf{K}) \tag{39}$$

In Eq. (39), $\mathcal{N}_{n,\mathcal{D}}(\lambda; \mathbf{K})$ represent the basis functions and are defined through the Cox-de Boor recursion formula (Boor, 1972). With regard to the time, it can be defined from λ as $t = t_f \lambda$. However, as already shown in (Spiller et al., 2017, 2016b), full advantage of the B-spline is taken approximating the trajectory as a curve, not as a function. In this case, the linear relationship between t and λ is no more valid and the time is evaluated as a B-spline. Once introduced the strictly-increasing time coefficient vector $\boldsymbol{b} = [b_{n,1} = 0, b_{n,2} > b_{n,1}, ..., b_{n,N_{\mathcal{P}-1}} = 1]$, the time is given as

$$t = t_f \mathcal{B}(\lambda; \boldsymbol{b}, \mathbf{K}) = t_f \sum_{n=0}^{N_{\mathcal{P}}-1} b_n \mathcal{N}_{n,\mathcal{D}}(\lambda; \mathbf{K}) \,. \tag{40}$$

Let $\mathcal{B}'_a = \partial \mathcal{B}(\lambda; \boldsymbol{a}, \mathbf{K})/\partial\lambda$ and $\mathcal{B}''_a = \partial^2 \mathcal{B}(\lambda; \boldsymbol{b}, \mathbf{K})/\partial\lambda^2$ be defined as

$$\mathcal{B}'_a = \sum_{n=0}^{N_P-2} a'_n \mathcal{N}_{n+1,\mathcal{D}-1}(\lambda; \mathbf{K}), \tag{41}$$

$$\mathcal{B}''_a = \sum_{n=0}^{N_P-3} a''_n \mathcal{N}_{n+2,\mathcal{D}-2}(\lambda; \mathbf{K}), \tag{42}$$

where $(\cdot)'$ is the derivative with respect to λ, and a'_n and a''_n are given by the following finite differences:

$$a'_n = \mathcal{D}\,\frac{a_{n+1} - a_n}{k_{n+\mathcal{D}} - k_n}\,, \quad a''_n = (\mathcal{D}-1)\frac{a'_{n+1} - a'_n}{k_{n+\mathcal{D}-1} - k_n}\,. \tag{43}$$

Eq. (41) and Eq. (42) can be obviouslly defined also for the first and second derivates of $\mathcal{B}(\lambda; \boldsymbol{b}, \mathbf{K})$. The time derivatives of the function ρ_n^N are then evaluated as

$$\begin{aligned} \dot{\rho}_{N;l} &= \frac{d\mathcal{B}(\lambda; \boldsymbol{a}, \mathbf{K})}{dt} = \frac{\partial \mathcal{B}(\lambda; \boldsymbol{a}, \mathbf{K})}{\partial \lambda}\frac{\partial \lambda}{\partial t} = \frac{\partial \mathcal{B}(\lambda; \boldsymbol{a}, \mathbf{K})}{\partial \lambda}\frac{\partial \lambda}{\partial \mathcal{B}(\lambda; \boldsymbol{b}, \mathbf{K})} = \frac{\mathcal{B}'_a}{\mathcal{B}'_b} \\ \ddot{\rho}_{N;l} &= \frac{d^2\mathcal{B}(\lambda; \boldsymbol{a}, \mathbf{K})}{dt^2} = \frac{\mathcal{B}''_a \mathcal{B}'_b - \mathcal{B}''_b \mathcal{B}'_a}{\mathcal{B}'^3_b}\,. \end{aligned} \tag{44}$$

Table 7: Orbital parameters of the chief satellite.

Parameter	a (km)	e	i (deg)	Ω (deg)	ω (deg)
Value	7000	0	45	0	0

To build a B-spline, a polyline defined over the control points $\mathbf{A}_j = [b_j, a_j]$, $j = 0, ..., N_{\mathcal{P}} - 1$, is introduced. In the sequel, *clamped* B-spline will be employed: in this case, the curve passes through $\mathbf{A}_0$ and $\mathbf{A}_{N_{\mathcal{P}}-1}$ and it is obtained using *non-uniform knot points* given by

$$\begin{aligned} k_n &= 0 && if \;\; 0 \leq n \leq \mathcal{D}\,, \\ k_n &= \frac{n - \mathcal{D}}{N_{\mathcal{P}} - \mathcal{D}} && if \;\; \mathcal{D} < n \leq N_{\mathcal{P}} - 1\,, \\ k_n &= 1 && if \;\; N_{\mathcal{P}} - 1 < n \leq m\,. \end{aligned} \tag{45}$$

Initial and final conditions on the l^{th} component of the state $x_l(t)$ are imposed by setting

$$\begin{gathered} a_0 = x_l(t_0)\,, \quad a_1 = a_0 + (b_1 - b_0)\dot{x}_l(t_0)\,, \\ a_{N_{\mathcal{P}}-1} = x_l(t_f)\,, \quad a_{N_{\mathcal{P}}-2} = a_{N_{\mathcal{P}}-1} - (b_{N_{\mathcal{P}}-1} - b_{N_{\mathcal{P}}-2})\dot{x}_l(t_f)\,. \end{gathered} \tag{46}$$

5. Numerical results

The optimal control problem explained in Sec. 4 is applied to the case of a controlled deputy and a non-cooperative chief spacecraft. Table 7 reports the orbital elements of the chief satellite, orbiting around the Earth on a circular inclined orbit with an orbital period of about $T_{orb} = 5828.52$ s. For this analysis, only the planar motion is taken into account, i.e., the motion along $Z_{\mathscr{L}}$ is not controlled.

Non-dimensional units are used; consequently, distances are divided by K_x, that is the initial relative distance between the two satellites, and the control is divided by $K_u = u_{max}$. The manoeuvre time is normalized by $K_t = \sqrt{K_x/u_{max}}$,

Table 8: Boundaries of the variables.

Variables	Boundaries
Coefficients for the displacements	$[-10, 10]$
Coefficients for the time vectors	$[0, 1]$
manoeuvre time	$[0.25, 1.25] \cdot T_{orb}/K_t$

and velocities are divided by $K_v = K_x/K_t$. The maximum value of the thrust has been set to $u_{max} = 5 \cdot 10^{-4}$ m/s^2. For what concerns with the B-spline, N_P is set equal to 8; whereas the trajectory is discretized over $N_t = 100$ points. The tolerance is computed by means of Eq. (30) with $N_\delta = 3$ for both the internal and external loops. The objective function is defined as

$$
\begin{aligned}
J &= \bar{t_f} + \sum_{l=1}^{2} \sum_{k=0}^{N_t} \eta_{l,k}(\boldsymbol{a}_l, t_k) + 100 \cdot N_{viol} \\
\eta_{l,k}(\boldsymbol{a}_l, t_k) &= \begin{cases} 0 & \text{if } \dfrac{|u_{l,k}(\boldsymbol{a}_l, t_k)|}{u_{max}} \leq 1\,, \\ \dfrac{|u_{l,k}(\boldsymbol{a}_l, t_k)|}{u_{max}} & \text{otherwise}\,. \end{cases}
\end{aligned}
\tag{47}
$$

Note that $\bar{t_f}$ is the normalized manoeuvre time and N_{viol} represent the number of violated constraints, which makes the algorithm prefer those CPs that violate a small number of constraints than the others. In particular, N_{viol} is set equal to 1 if u_x or u_y exceeds the maximum admitted value u_{max}, otherwise it is set equal to 0. Using improved B-splines, the vector $\mathbf{X}_i$, associated to each CP of IMCSS, is made up of 21 variables. Indeed, $2(N_P - 4)$ and $2(N_P - 2)$ are the parameters needed for the B-spline approximation respectively for the trajectory and the time (along both dimensions), and the last variable represents the time of the manoeuvre. The boundaries chosen for the variables are shown in Table 8

Initial and final state vectors are respectively: $[x(0), y(0), \dot{x}(0), \dot{y}(0)] = [-1$ km, 1 km, 0 km/s, $-2\omega x(0)$ km/s$]$, where ω is the mean motion of the chief satellite, and $[x(f), y(f), \dot{x}(f), \dot{y}(f)] = [0$ km, 0 km, 0 km/s, 0 km/s$]$. According to the initial conditions, $K_x = 1.4142$ km, $K_t = 1681.793$ s and $K_v = 8.40896 \cdot 10^{-4}$ km/s.

The results of the optimization are shown in Fig 5. As it can be seen from the control policy, the control approximates very well the bang-bang control, which is proven to be the ideal optimal control by the Pontryagin Maximum Principle (PMP) applied to the Hamiltonian written for the SS model (see (Kirk, 2012) for further details). The computed manoeuvre time is 3921.74 s, which is lower than previous results in literature (in (Parente et al., 2018) the manoeuvre time for this problem was shown to be 3940 s, by using the PSO, and 3930 s, by means of DE). This comparison demonstrates the effectiveness of the proposed IMCSS algorithm and its competitiveness with respect to more common algorithms in the field.

Another simulation was performed to demonstrate the effect of the boundaries constraints handling technique also for this optimal control problem. Let the upper bound of the manoeuvre time be wrongly defined as $[0.25, 0.5] \cdot T_{orb}/K_t$. The optimal time previously computed is not within this range. Table 9 illustrates one case of the history of the optimization procedure. c_{UB} and c_{LB} indicate the counters for the violation of the lower and upper bounds of the manoeuvre time. The last two columns show the history of the lower and upper bound of the manoeuvre time in dimensionless unit. For the first three external

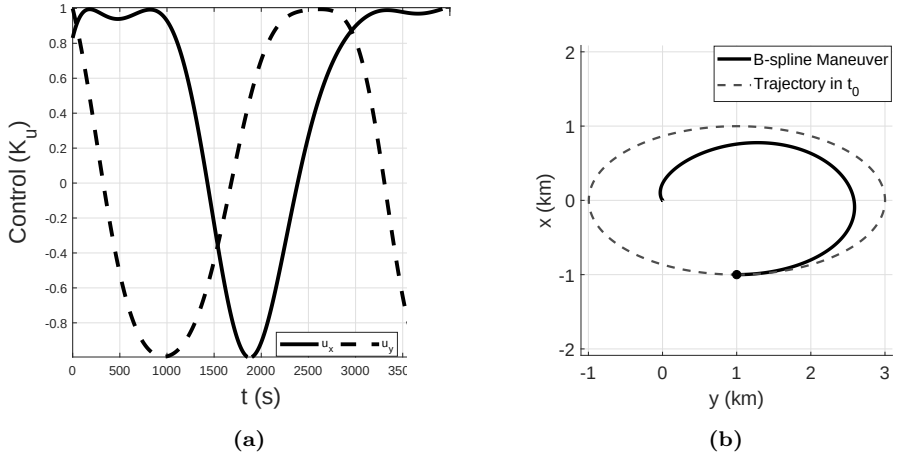

Figure 5: Normalized control policy (a) and trajectory (b).

loops, IMCSS is able to decrease the objective function, but the constraint on the maximum control is violated. Up to the third external loop, the sixth column indicates that the boundary is not violated, although the counter shows an increasing trend due to the fact that IMCSS is decreasing the objective function acting only on the manoeuvre time (but not on the control constraint). Between the third and the fourth loop, the performance index decreases only by 4 unities; this means that IMCSS cannot decrease anymore the objective function trying to fulfill at the same time the control constraint. This is an evidence of the fact that something is going wrong; in fact, if one looks at the counter of the boundary violation at the end of the loop, it is greater than the maximum permitted. This violation causes IMCSS to understand that the boundaries of the manoeuvre time have been bad-defined by the user and thus it enlarges the upper boundary. As can be seen, from the fifth loop to the end of the optimization process, IMCSS is able to change the search direction toward the real optimal solution. The final normalized manoeuvre time is $\bar{t}_f = 2.34884$, which corresponds to 3950.27 s (very close to the actual optimum). Hence, this example has shown again the ability of IMCSS to converge to a very good result, correcting at the same time the error made by the user.

Table 9: Results of the boundary constraints handling technique for the reconfiguration manoeuvre with $N_{CP} = 50$.

G_K	$G_{k,f}$	J_g	δ	C_{max}	c_{UB}	c_{LB}	UB	LB
1	450	745.35684	0.08254	2250	1517	70	1.73283	0.866415
2	450	662.02823	0.42898	2250	1925	61	1.73283	0.866415
3	450	626.86848	0.48313	2250	2108	60	1.73283	0.866415
4	450	622.39754	0.15988	**2250**	**2278**	98	1.73283	0.866415
5	450	3.06361	0.03517	2250	258	117	**17.3283**	0.866415
6	450	2.86168	0.00040	2250	167	93	17.3283	0.866415
7	450	2.42161	0.00314	2250	230	108	17.3283	0.866415
8	2250	2.34884	0.00550	11250	106	76	17.3283	0.866415

6. Conclusions

An Improved Magnetic Charged System Search (IMCSS) algorithm is developed to improve the performances of the original MCSS. To this aim, some useful tools are introduced, such as the chaotic local search and the boundary constraints handling technique. Moreover, significant steps towards the self-adaptivity of the algorithm are carried out. In fact, the algorithm is able to dynamically change its internal coefficients and automatically provide suitable initial parameters as functions of the lower and upper bounds. All these concepts can also be adopted to improve and speed up the convergence of other metaheuristic algorithms and the potentialities of these strategies are shown. Indeed, the analysis with benchmark functions, which have different features and order of magnitudes, and classical engineering problems shows good results compared to the ones reported in literature. The paper demonstrates that the boundary constraint handling technique is effective when the lower and upper bounds of the variables are bad-defined. In addition, it has been demonstrated that the chaotic local search is a useful tool for reaching the global optimum in a faster way, and it has proved to be successful in most of the cases. Moreover, the entropy of the system is successfully used to ensure a better convergence by increasing the initialized number of particles during the optimization process. Furthermore, the space application dealing with time optimal manoeuvre for the docking of two satellites is studied imposing an inverse dynamics approach. The computed optimal time of the manoeuvre indicates an improvement with respect to the literature. Since the optimization problems faced in this work are very different from each other, this paper represents a good proof of having chosen suitable mathematical laws for the parameters of the IMCSS algorithm. To conclude, the proposed algorithm is demonstrated to be efficient and reliable to solve optimization problems, including those related to space engineering.

References

Arora, J. S. (1989). *Introduction to optimum design*. McGraw-Hill, New York.

Bandaru, S., & Deb, K. (2016). *Metaheuristic Techniques*. Published in Decision Sciences: Theory and Practice Press, CRC, Taylor & Francis Group.

Bartz-Beielstein, T., Lasarczyk, C., & Preu, M. (2005). Sequential parameter optimization. *IEEE Congress Evol. Comput. 1*, (pp. 773–780).

Belegundu, A. D. (1983). A study of mathematical programming methods for structural optimization. ph.d. thesis, department of civil and environmental engineering, university of iowa, iowa., .

Birattari, M., Sttzle, T., Paquete, L., & Varrentrapp, K. (2002). A racing algorithm for configuring metaheuristics. *Proceedings of the Genetic and Evolutionary Computation Conference 2*, (pp. 11–18).

Boor, C. D. (1972). On calculating with b-splines. *Journal of approximation theory*, *6*, 50–62. doi:10.1016/0021-9045(72)90080-9.

Boyarko, G., Yakimenko, O., & Romano, M. (2011). Optimal rendezvous trajectories of a controlled spacecraft and a tumbling object. *Journal of Guidance, Control, and Dynamics*, *34*, 1239–1252. doi:10.2514/1.47645.

Civicioglu, P., Besdok, E., Gunen, M. A., & Atasever, U. H. (2018). Weighted differential evolution algorithm for numerical function optimization: a comparative study with cuckoo search, artificial bee colony, adaptive differential evolution, and backtracking search optimization algorithms. *Neural Computing and Applications*, (pp. 1–15). doi:10.1007/s00521-018-3822-5.

Clerc, M. (2010). *Particle swarm optimization*. London, UK: ISTE Ltd.

Coello, C. A. C. (2000). Use of a self-adaptive penalty approach for engineering optimization problems. *Computers in Industry*, *41*, 113–127.

Coello, C. A. C., & Montes, E. M. (2002). Constraint-handling in genetic algorithms through the use of dominance-based tournament selection. *Advanced Engineering Informatics*, *16*, 193–203.

David, V., & McClain, W. D. (2007). *Fundamentals of Astrodynamics and Applications*. Microcosm Press.

Deb, K. (1991). Optimal design of a welded beam via genetic algorithms. *AIAA journal*, *29*, 2013–2015.

Deb, K. (1997). Geneas: A robust optimal design technique for mechanical component design. In *Evolutionary algorithms in engineering applications* (pp. 497–514). Springer.

Dorigo, M., Maniezzo, V., & Colorni, A. (1996). Ant system: optimization by a colony of cooperating agents. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, *26*, 29–41.

Fliess, M., Lévine, J., Martin, P., & Rouchon, P. (1995). Flatness and defect of non-linear systems: introductory theory and examples. *International journal of control*, *61*, 1327–1361.

Goldberg, D. (1989). *Genetic algorithms in search, optimization, and machine learning*. Reading, Massachusetts, USA: Addison-Wesley.

Guo, Z., Huang, H., Deng, C., Yue, X., & Wu, Z. (2015). An enhanced differential evolution with elite chaotic local search. *Computational intelligence and neuroscience*, *2015*, 6.

Harrison, K., Engelbrecht, A., & Ombuki-Berman, B. (2017). Self-adaptive particle swarm optimization: a review and analysis of convergence. *Swarm Intelligence*, *12*, 187–226. doi:10.1007/s11721-017-0150-9.

Hartmann, A. H., & Rieger, H. (2003). *Optimization Algorithms in Physics*. WileyVCH. doi:10.1002/3527600876.

He, Q., & Wang, L. (2007). An effective co-evolutionary particle swarm optimization for constrained engineering design problems. *Engineering applications of artificial intelligence*, *20*, 89–99.

Holland, J. (1975). *Adaptation in Natural and Artificial Systems*. University of Michigan Press, Ann Arbor, MI.

Ji, J., Gao, S., Wang, S., Tang, Y., Yu, H., & Todo, Y. (2017). Self-adaptive gravitational search algorithm with a modified chaotic local search. *IEEE Access*, *5*, 17881–17895. doi:10.1109/ACCESS.2017.2748957.

Jia, D., Zheng, G., & Khan, M. K. (2011). An effective memetic differential evolution algorithm based on chaotic local search. *Information Sciences*, *181*, 3175–3187.

Kalyanmoy, D. (2004). *Optimization for Engineering Design: Algorithms and Examples*. PHI Learning.

Kannan, B., & Kramer, S. N. (1994). An augmented lagrange multiplier based method for mixed integer discrete continuous optimization and its applications to mechanical design. *Trans. ASME J. Mech. Des.*, *116*, 318–320.

Kaveh, A., Khanzadi, M., Moghaddam, M. R., & Rezazadeh, M. (2018). Charged system search and magnetic charged system search algorithms for construction site layout planning optimization. *Periodica Polytechnica Civil Engineering*, *62*, 841–850.

Kaveh, A., Mirzaei, B., & Jafarvand, A. (2014). Shape-size optimization of single-layer barrel vaults using improved magnetic charged system search. *Int J Civil Eng*, *12*, 447–465.

Kaveh, A., Mirzaei, B., & Jafarvand, A. (2015). An improved magnetic charged system search for optimization of truss structures with continuous and discrete variables. *Applied soft computing*, *28*, 400–410.

Kaveh, A., Share, M. M., & Moslehi, M. (2013). Magnetic charged system search: a new meta-heuristic algorithm for optimization. *Acta Mechanica*, *224*, 85–107.

Kaveh, A., & Talatahari, S. (2009). Particle swarm optimizer, ant colony strategy and harmony search scheme hybridized for optimization of truss structures. *Computers and Structures*, *87*, 267–283. doi:10.1016/j.compstruc.2009.01.003.

Kaveh, A., & Talatahari, S. (2010a). An improved ant colony optimization for constrained engineering design problems. *Engineering Computations*, *27*, 155–182.

Kaveh, A., & Talatahari, S. (2010b). A novel heuristic optimization method: charged system search. *Acta Mechanica*, *213*, 267–289.

Kaveh, A., & Zolghadr, A. (2014). Magnetic charged system search for structural optimization. *Periodica Polytechnica Civil Engineering*, *58*, 203–216.

Kim, Y., & Spencer, D. (2002). Optimal spacecraft rendezvous using genetic algorithms. *Journal of Spacecrafts and Rockets*, *39*, 859–865. doi:10.2514/2.3908.

Kirk, D. E. (2012). *Optimal Control Theory: An Introduction*. Dover Books on Electrical Engineering. Dover Publications.

Kumar, Y., Gupta, S., Kumar, D., & Sahoo, G. (2016). A clustering approach based on charged particles. *Optimization Algorithms-Methods and Applications*, (pp. 245–263).

Lee, K., & Geem, Z. (2004). A new structural optimization method based on the harmony search algorithm. *Computers and Structures*, *82*, 781–798. doi:10.1016/j.compstruc.2004.01.002.

Louembet, C. (2007). Design of algorithms for satellite slew manoeuver by flatness and collocation. *Proceedings of the American Control Conference, IEEE Publ., Piscataway, NJ*, (pp. 3168–3173). doi:10.1109/ACC.2007.4282459.

Mezura-Montes, E., & Coello, C. A. C. (2008). An empirical study about the usefulness of evolution strategies to solve constrained optimization problems. *International Journal of General Systems*, *37*, 443–473.

Mohajan, H. K., Islam, J. N., & Moolio, P. (1953). *Optimization and Social Welfare in Economics*. LAP LAMBERT Academic Publishing.

Nannen, V., & Eiben, A. (2007). Relevance estimation and value calibration of evolutionary algorithm parameters. *Int. Joint Conf. Artif. Intell. 7*, (pp. 975–980).

Parente, D., Spiller, D., & Curti, F. (2018). Time-suboptimal satellite formation maneuvers using inverse dynamics and differential evolution, . *41*, 1108–1121. doi:10.2514/1.G003110.

Pontani, M., & Conway, B. (2013). Optimal finite-thrust rendezvous trajectories found via particle swarm algorithm. *Journal of Spacecrafts and Rockets*, *50*, 1222–1234. doi:10.2514/1.A32402.

Ragsdell, K., & Phillips, D. (1976). Optimal design of a class of welded structures using geometric programming. *Journal of Engineering for Industry*, *98*, 1021–1025.

Rashedi, E., Nezamabadi-Pour, H., & Saryazdi, S. (2009). Gsa: A gravitational search algorithm. *Information Science*, *179*, 2232–2248.

Ross, I., & Fahroo, F. (2002). A perspective on methods for trajectory optimization. *Proceedings of the AIAA/AAS Astrodynamics Specialist Conference and Exhibit, Guidance, Navigation, and Control and Co-Located Conferences, AIAA Paper 2002-4727*, . doi:10.2514/6.2002-4727.

Sandgren, E. (1988). Nonlinear integer and discrete programming in mechanical design. In *Proceeding of the ASME Design Technology Conference* (pp. 95–105).

Schweighart, S., & Sedwick, R. (2001). A perturbative analysis of geopotential disturbances for satellite cluster formation flying. *Proceedings of the 2001 IEEE Aerospace Conference*, *2*, 1001–1019. doi:10.1109/AERO.2001.931281.

Schweighart, S., & Sedwick, R. (2002). High-fidelity linearized J2 model for satellite formation flight. *Journal of Guidance, Control,and Dynamics*, *25*, 1073–1080. doi:10.2514/2.4986.

Spiller, D., Ansalone, L., & Curti, F. (2016a). Particle swarm optimization for time-optimal spacecraft reorientation with keep-out cones. *Journal of Guidance, Control, and Dynamics*, *39*, 312–325. doi:10.2514/1.G001228.

Spiller, D., Curti, F., & Circi., C. (2017). Minimum-time reconfiguration maneuvers of satellite formations using perturbation forces. *Journal of Guidance, Control, and Dynamics*, *5*, 1130–1143. doi:10.2514/1.G002382.

Spiller, D., Melton, R., & Curti, F. (2016b). Inverse dynamics particle swarm optimization applied to constrained minimum-time maneuvers using reaction wheels. *67th International Astronautical Congress (IAC 2016), International Astronautical Federation (IAF), IAC-16, C1,8,1, x31975*, . doi:10.2514/1.G001228.

Storn, R., & Price, K. (1997). Differential evolution - a simple and efficient heuristic for global optimization over continuous spaces. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, *11*, 341–359.

Vecek, N., Mernik, M., Filipic, B., & Crepinek, M. (2016). Parameter tuning with chess rating system (crs-tuning) for meta-heuristic algorithms. *Information Sciences*, *372*, 446–469. doi:10.1109/ACCESS.2017.2748957.

Wang, S., & Zheng, C. (2012). A hierarchical evolutionary trajectory planner for spacecraft formation reconfiguration. *IEEE Aerospace and Electronic Systems Magazine*, *48*, 279–289. doi:10.1109/TAES.2012.6129635.

Yang, X. (2010). *Engineering Optimization: An Introduction with Metaheuristic Applications*. Wiley.

Yang, X. (2018). *Optimization Techniques and Applications with Examples*. Wiley.

Zhang, D., Song, S., & Duan, G. (2008). Fuel and time optimal transfer of spacecrafts rendezvous using lamberts theorem and improved genetic algorithm. *Proceedings of the 2008 2nd International Symposium on Systems and Control in Aerospace and Astronautics, ISSCAA 2008, IEEE Publ., Piscataway, NJ*, (pp. 1–6). doi:10.1109/ISSCAA.2008.4776390.