A SYSTEMATIC-HEURISTIC APPROACH FOR SPACE TRAJECTORY DESIGN

Massimiliano Vasile
European Space Research & Technology Centre
(ESA/ESTEC),Keplerlaan 1,2200 AZ,Noordwijk ZH, The Netherlands
Ph.+31-71-565-4570
Fax+31-71-565-8018
Massimiliano.Vasile@esa.int

Abstract

In this paper a novel algorithm combining a systematic and a heuristic method for global optimization is proposed for space trajectory design. For the systematic part of the algorithm a branching technique is used while a particular implementation of evolution programming forms the core of the heuristic part. The idea is to use a limited population evolving for a small number of generations, according to some specific evolution rules, in subregions of the solution space defined by a branching procedure. On the other hand the branching rules are functions of the outcome from the evolution optimization. The proposed combined systematic-heuristic global optimization performs quite well on the cases analyzed in this paper, suggesting the possibility of more complex applications.

Keywords: global optimization, space trajectory design, evolution programming, branching methods

INTRODUCTION

A meaningful part of the mission design process consists of designing the trajectory. Traditionally this task has been accomplished using gradient methods, optimal control theory or mathematical tools specifically dedicated to each particular problem. Anyway, all these approaches can be generally classified as local optimization methods where the term optimization is intended not just for finding the minimum or the maximum but more in general for finding a solution. Since the problem is generally highly non-linear and not necessary differentiable in entire the solution space, a significant part of the job is to formulate appropriately the problem to make it amenable to a solution using local optimization tools and to produce a reasonably good initial guess. In fact, it is likely that, despite the global convergence properties of many software tools (like most of NLP solvers available at present), the analysts find a local minimum every time they seek for a solution, eventually finding the global optimum. This is quite a time consuming process that can lead even to a non-satisfactory result. Furthermore, due to the relatively poor robustness of some approaches (like indirect methods) a good initial guess is often quite hard to find. Finally, a complete and sophisticated formulation of the problem can generally include discontinuities or integer variables or non-differentiable functions impossible to handle with standard gradient methods.

The importance of having an effective and efficient global optimization approach (well known, understood and studied in many other fields like chemistry, biology and electronic engineering[1]) is recently emerging even in the space field with studies on procedure and optimization methods to procure a solution or even just a first guess solution to complex

problems such as the design of lowthrust propulsion trajectories[2,3], the design of WSB (Weak Stability Boundaries)[4] transfers, the characterization of geocentric orbits in the rescricted three-body problem[5], the analysis of trajectories involving multiple swing-bys or a combination of swing-bys and low thrust propulsion[6,7].

Most global optimization techniques can be classified in three main groups: approximation methods, systematic methods and heuristic, or stochastic methods. Approximation methods are not considered here because they need a suitable approximation to the original problem.

In this paper,instead, a mixed approach combining a systematic technique such as branching and a particular implementation of evolution programming[8],which falls among heuristic methods, is proposed to solve some space trajectory design problems. The basic idea is to use EP to explore subregions of the solution space defined by a branching procedure. The outcome of the EP exploration is then used to select promising branches, to prune non-promising ones and to define future branching rules. The particular implementation of EP is characterized by a migration operator, that guides individuals toward promising areas of the solution space, and a filter operator (in place of common selection operators) ranking families of potentially interesting individuals. Furthermore, the algorithm uses a combined integer-floating point encoding of the individuals allowing the solution of mixed integer-nonlinear programming problems.

The effectiveness of the proposed algorithm (EPIC) is demonstrated on two problems of space trajectory design.

PROBLEM FORMULATION

Optimization problems in trajectory design can be either unconstrained or constrained; in their general form they can be written as:

$$\begin{aligned} &\min f(\mathbf{y}) \\ &\mathbf{b}^l \le \mathbf{C}(\mathbf{y}) \le \mathbf{b}^u \\ &\text{with} \quad \mathbf{y} \in D \end{aligned} \tag{1}$$

where f is a scalar nonlinear function of a multidimensional vector $\mathbf{y}$ defined within the domain D. The domain D is a hypercube defined by the upper and lower bounds on the components of the vector $\mathbf{y}$:

$$y_i \in [b_i^l, b_i^u] \tag{2}$$

The vector $\mathbf{C}(\mathbf{y})$ is formed by all nonlinear constraint functions of the vector $\mathbf{y}$. If problem (1) is twice continuously differentiable and presents a single solution, i.e. only one vector $\mathbf{y}$ in the domain D minimizes f and satisfies $\mathbf{C}$, a nonlinear programming method like sequential quadratic programming (SQP) can be efficiently used. This means implicitly that the problem must be formulated properly and cannot contain non-differentiable quantities. However even in this case the problem may present more than one solution within the required domain D.

If the problem is either non-differentiable, i.e. no gradient method can be applied, or more than a solution is expected, a global optimization method must be considered. The idea is to perform an extensive search of the solution space D looking for possible solutions to problem (1). In this respect the interest could be more to find a number of good initial guesses for the nonlinear programming solver, than finding the global optimum with a high level of accuracy.

It is common to classify methods for global optimization in three categories: approximation methods, heuristic methods and systematic methods.

Approximation methods transform the original problem by means of suitable approximations into a simpler global optimization problem that is more tractable. Once a solution for the approximated problem is found a local optimization method can be used.

Heuristic methods contain all methods that cannot be proven to find a global optimum with a predictable amount of work. Most stochastic methods are in this class. For them, it is sometimes possible to prove convergence with probability arbitrarily close to 1 but with a number arbitrarily large of function evaluations. The simplest heuristic method is multiple random start, consisting of picking random starting points and performing local optimizations from these points. Most heuristics can be regarded as techniques devised to speed up this process of local/global search by picking the points more carefully.

Systematic methods contain all methods that (in exact arithmetic) are guaranteed to find the global optimum with a predictable (deterministic) amount of work. The bound on the amount of work is anyway quite high: exponential in the problem characteristics. The simplest systematic method for bound constrained problems is grid search where all points on finer and finer grids are tested and the best point on each grid is used as starting point for local optimization. The number of grid points grows exponentially with the dimensions of the problem and so does the amount of work. More efficient systematic methods generally combine branching techniques with one or several local optimization procedures, convex analysis, interval analysis and constraint logic. Even though systematic methods are generally more reliable then heuristics they need, to be efficient, some level of insight into the problem and the structure of the objective function. If the problem is represented by a black box then they may not find the global optimum in a reasonable amount of time. This is understandable if we look at the density theorem (Törn and Zilinskas 1989), which states that any method based on local information only, that converges for every continuous f to a global minimizer of f in a feasible domain D must produce a sequence of points $y^1, y^2, y^3, \ldots$ that is dense in D.

A well known stochastic method is represented by Genetic Algorithms (GA) that make use of analogies to biological evolution by allowing mutations and crossing over among candidates for good local optima in the hope to derive even better ones. The original concept of Genetic Algorithms is to encode a potential solution (individual) of the problem under study, in the form of a binary string in which each binary number represents a chromosome of the "DNA" (or genotype) of the solution (or phenotype). More sophisticated genetic algorithms make use of the data structure of the problem to encode the individual in the more appropriate way. For example if the problem contains only floating point variables a floating point encoding is more accurate and efficient than a binary encoding. In general all methods that resort to some heuristics concepts derived from biological evolution can be defined as evolution programming methods. An interesting concept is represented by niching-GA[9]. The basic concept is that in nature different species can exploit different niches in the environment. This translates in the formation of subpopulations with each subpopulation specialized at a subtask of the problem or exploring a subregion. Subpopulations can compete as in pure GA or cooperate.

Among systematic methods there are some that split the solution domain on the base of some local information. Each time the domain is split, a number of new branches are created, each branch corresponds to a further exploration of the solution space and each subdomain represents a node that can be expanded and explored further. Therefore a branching scheme generates a sequence of rooted trees of boxes whose leaves cover the feasible set. If the diameter of all the boxes and all leaves converge to zero, convergence of the algorithm is straightforward.

Proposed optimization approach is composed of a stochastic step and of a systematic step. The stochastic step is performed using evolution programming and is meant to obtain information on the possible presence of optima in a subdomain $D_l \subseteq D$. On the other hand the systematic step is performed through a branching approach and is used to partition the domain D into subdomains D_l. where the presence of an optimum is expected. Each subdomain may or may not contain the global optimum but the systematic exploration and the qualification of each subdomain on the

base of the best solution found and the volume of the subdomain, allows finding a number of optima and eventually the global one. This particular hybridization can be seen as a form of niching since populations evolving in subregions form different species.

EVOLUTION PROGRAMMING

Present implementation of evolution programming is based on four fundamental operators: mutation, migration, mating and filtering.

Problem Encoding. Each solution **y** is represented by a string containing in the first *m* components integer values and in the remaining *s* components real values. This particular encoding allows the treatment of problems with a mixed integer-real data structure. It should be noticed that, as a consequence, all operators are designed to handle both real and integer numbers at the same time.

Mutation. Mutation operates in three different ways: generates a random number, taken from a gaussian distribution, within the domain *D* or within each subdomain D_l, for each component of **y**; generates a symmetric perturbation of a selected component y_i with respect to its original value within an interval in a neighborhood of **y**. A third mutation scheme exchanges a random componet of the individual with one of the bounds.

Mating. The mating procedure takes two individuals and generates one or two children mixing the genotypes of the two parents. Four schemes are used to mate individuals:

- Single point crossover which simply exchange part of the genes between the two parents
- Blending, also known as arithmetic crossover, which generates a new individual with an interpolation of the two parents: chosen two individuals $\mathbf{y}^1$ and $\mathbf{y}^2$ and a random number α, the resulting child will be given by:
$$\mathbf{y}^3 = \alpha\mathbf{y}^2 + (1-\alpha)\mathbf{y}^1 \tag{3}$$
- Extrapolation generates a new individual on the side of the best individual between the two parents at a distance from the best parents equal to the vector connecting the two parents:
$$\mathbf{y}^3 = \alpha(\mathbf{y}^2 - \mathbf{y}^1) + \mathbf{y}^2 \tag{4}$$
- Second order extrapolation mating generates a child using two parents and the child generated with an extrapolation mating. If **p** is the vector difference between $\mathbf{y}^1$ and $\mathbf{y}^3$ and f^1, f^2, f^3 are the fitness values for the three individuals $\mathbf{y}^1$, $\mathbf{y}^2$,$\mathbf{y}^3$ respectively, then a second order one-dimensional model of the fitness function is built and the new child is generated taking the minimum of the resulting parabola (see Figs. 1 and 2):
$$\mathbf{y}^4 = \mathbf{y}^1 + \mathbf{p}\chi_{\min} \tag{5}$$
$$f_{\min} = a(\mathbf{y}^1, \mathbf{y}^2, \mathbf{y}^3)\chi^2_{\min} + b(\mathbf{y}^1, \mathbf{y}^2, \mathbf{y}^3)\chi_{\min} + f(\mathbf{y}^1) \tag{6}$$

The mating operator is used also to prevent an undesirable effect of migrations: if more than one individual is in the basin of attraction of the same solution, it is likely that all of them will move toward the same point with a resulting waste of resources. Therefore, if two or more individuals are colliding (intersecting their migration regions) a repelling mechanism is activated which mates the worse individual (between two colliding) with the boundaries of the subdomain D_l: each component of the selected individual is blended with the value of the furthest bound, projecting the individual into a random point within D_l, according to the following relation:
$$y_i^2 = \alpha b_i + (1-\alpha)y_i^1 \tag{7}$$

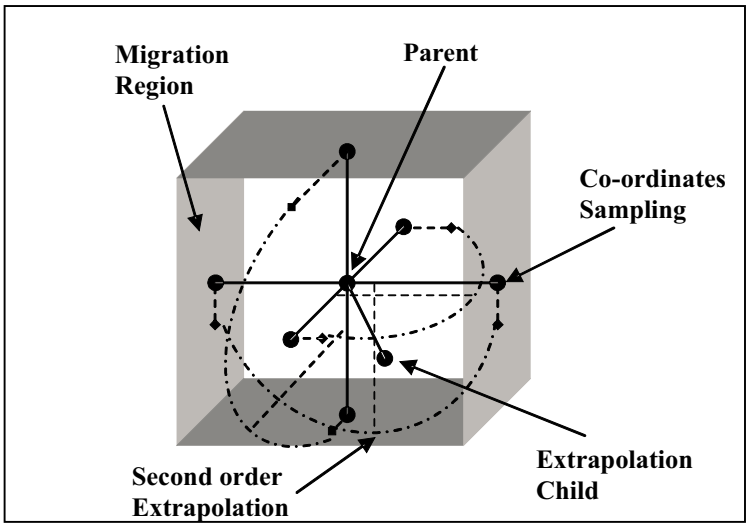

Figure 1. Deterministic criterion for the generation of a migrating subpopulation

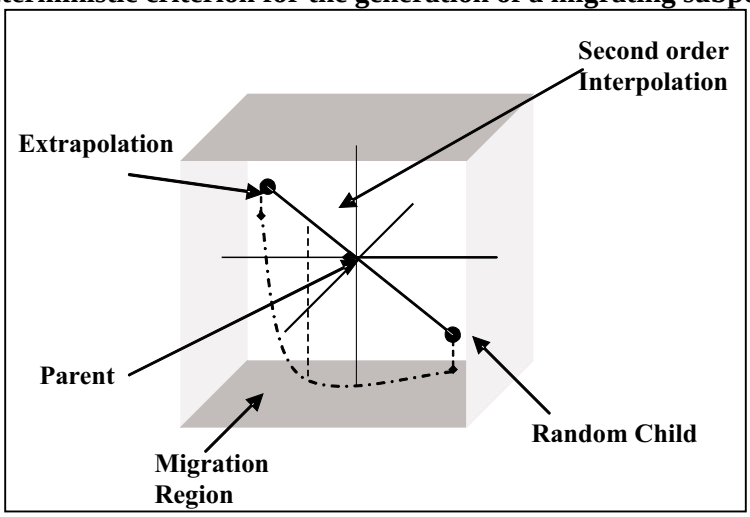

Figure 2. Stochastic criterion for the generation of a migrating subpopulation

Migration

The migration process generates a subpopulation in a neighborhood of an individual **y** called principal individual**.** Then the best individual of the subpopulation, if better than the parent, survives to the next generation instead of the parent. The subpopulation is generated using either a stochastic or a deterministic procedure. The deterministic procedure generates a child along the coordinates on the boundary of the migration region. Then an extrapolation mating produces a second child toward the most promising boundary. If the second and the first children are not the same a third one is generated using a quadratic interpolation of the two previous children and the parent. The same scheme is repeated for all the coordinates and all the children generated with the quadratic interpolation are then collected and used to generate an extrapolation child whose components are the components of all the second order children (see Fig.1).

The stochastic procedure samples the migration region generating a child randomly then an extrapolation mating is performed on the side of the best individual. The two resulting children and the parents are then used to build a quadratic model and a third child is generated using the second order extrapolation mating. The procedure is repeated until a number of children equal to the number of coordinates have been generated (see Fig. 2).

The micro population is generated within a migration region in a neighborhood of a principal individual; for an n-dimensional problem the migration region is a hyper parallelepiped $\mathbf{S}=S_1xS_2...xS_n \subseteq D_l$, where S_i is a subinterval containing the value of the component y_i. Each subinterval S_i is asymmetric, allowing the migration to depart from one of the boundaries. Furthermore the migration region **S** contracts or expands according to a migration radius ρ.

The migration radius

If f_k^j is the fitness value associated to an individual j at generation k and $f_{k+1}{}^j$ the fitness value associated to the same individual at generation $k+1$, we define the differential improvement as:

$$df^j = f_{k+1}^j - f_k^j \tag{8}$$

The migration radius is defined as the ratio between the value of the distance form the boundary $\mathbf{b}^j$ of the migration region of the j-th individual and the value of the distance from the correspondent boundary **b** of the domain D:

$$\rho^j = \frac{b_i^j - y_i^j}{b_i - y_i^j} \tag{9}$$

During convergence the migration radius is reduced or enlarged depending on the fitness of the individuals inside the migration region and the differential improvement of the principal individual from one generation to another: if none of the children of the subpopulation is better than the parent the radius is reduced. Different from a previous implementation[15], where the migration radius was a function of the differential improvement, in this implementation a slower contraction of the migration region has been implemented computing the radius according to:

$$\rho^j = \begin{cases} \max([1e-8, \delta y_{\min}]) & \text{if } \delta y_{\min} \geq \varepsilon\rho^j \\ \varepsilon\rho^j & \text{if } \delta y_{\min} < \varepsilon\rho^j \end{cases} \tag{10}$$

where ε has been set to 0.5 and δy_{min} is the distance of the best childe $\mathbf{y}^*$, among the ones in the migration region, from the parent j, normalized with respect to the dimensions of the migration region:

$$\delta y_{\min} = \sqrt{\sum_{i=1}^{n} \left(\frac{y_i^* - y_i^j}{S_i^j} \right)^2} \tag{11}$$

where, for individual j and for dimension i, S^j_i is the difference between the value of the upper bound and of the lower bound and the summation is over non-zero dimensions.

Now, if from generation k to generation $k+1$ the differential improvement increases, then the migration radius is recomputed according to the prediction:

$$\rho_{k+1}^j = \rho_k^j \eta \log(e-1+j) \tag{12}$$

where η is equal to 2 in this implementation. It should be noticed that the value of ρ^j_{k+1} depends on the ranking of the associated individual: the migration radius of a bad individual is therefore larger than the best individual allowing a bigger mutation. In fact if a principal individual is converging to a solution it is not desirable to move it too far from its position.

For integer numbers migration operates in the same way but the migration regions and migration radius are generated and treated differently. In particular ρ_{imin} is 1 and ρ is defined as:

$$\rho_{k+2}^j = \min[\text{int}(\log(2+j)\Delta f_{mig}{}^j), \rho_{i\min}] \tag{13}$$

The migration region is therefore contracted differently for real and for integer variables allowing a better spatial exploration.

Filtering

Instead of traditional selection mechanisms based on fitness here a permanent population of n individuals is maintained from one generation to another. Each individual has a chance to survive provided that it remains inside the filter. The filter ranks all the individuals on the basis of their fitness from the best to the worst. All the individuals with a fitness worse than a given threshold are hibernated (i.e. no operator is applied) while migration is applied to all individuals within the filter. This allows each of the individuals within the filter to evolve toward a different local optimum. The filter basically operates a simple sorting procedure but, since individuals in the upper part of the filter are strongly mutated, it is likely that they are replaced by quite different new individuals coming from the associated subpopulation and migration.

Not only does the position in the filter determine the level of mutation of the individual but also the mating process is influenced by the ranking. In fact, mating is operated on all the individuals in the upper part of the filter combining them with individuals in the lower part of the filter. After mating, the resulting children survive according to their predicted position in the filter. Therefore, in the pure EP step, migration is used to explore locally the solution space and two mechanisms are used for global exploration: mating and repulsion of clustering individuals in the filter. It should be noticed that if several minima are clustered the mixed systematic-stochastic generation of the subpopulation should guarantee anyway to find locally the best minimum of the cluster.

COMBINED EVOLUTION BRANCHING

Even though the filter increases the chances of finding several optima and eventually the global one, convergence is not guaranteed due to the stochastic nature of the process. Therefore, a systematic step is taken on the basis of the output of the evolution algorithm. The initial domain $D_0 \equiv D$ is partitioned generating a number of subdomains D_l. Each subdomain is then qualified and explored further according to its qualification.

The partitioning, or branching, process begins taking the worst individual, which is out of the filter, and cutting D_0 into L subdomains, corresponding to L potentially new branches (or nodes). Each one of the L nodes may or may not contain an individual coming from the previous step of evolution and the volume of the node depends on the position of the cutting point (a special mechanism prevents cuts too close to a boundary moving the cutting point to the middle of the interval). For each node D_l the ratio between the relative number of individuals and the relative volume is computed and the resulting quantity defines how necessary further exploration of the node is:

$$\varpi_{D_l} = \frac{\sum_{D_l} j}{\sum_{D} j} \Bigg/ \sqrt[n]{\frac{V_{D_l}}{V_D}} \qquad l=1,\ldots,L \tag{14}$$

where the volumes V_{Dl} and V_D are computed taking only edges with a non-zero dimension. This quantity is then added to a fitness φ_{Dl} defined as:

$$\varphi_{D_l} = \begin{cases} \dfrac{\dfrac{1}{J}\sum_{j=1}^{J} f_j - f_{best}}{f_{worst} - f_{best}} & if\ i \neq 0 \\ 1 & otherwise \end{cases} \tag{15}$$

where J is the number of individuals in domain D_l. The node is then qualified by the quantity:

$$\psi_{D_l} = \sigma\varpi_{D_l} + (1-\sigma)\varphi_{D_l} \tag{16}$$

where σ is the weighting factor that weights how reliable the result coming from the evolution step is considered. If σ is 0, only the nodes with low fitness are explored because the EP

algorithm is considered reliable enough to explore exhaustively the domain D without leaving any region unexplored, on the other hand if σ is 1 the result from the EP algorithm is considered to be not reliable due to a premature convergence or to a poor exploration of the solution space. It is clear in fact that a node with a large number of individuals with high fitness (at the top of the ranking scale) has a high probability to contain the global optimum if the EP has explored exhaustively the solution space, on the other hand if the volume of the node is large and the number of individuals low it is likely that the EP step was not able to evaluate sufficiently the solution space in this region and therefore the node needs still to be explored in the future. Now every time a node D_l is subdivided into other Q subnodes only the most promising pair is taken into account. If ψ_{Dl} is used to select the most promising ones among all L subdomains, the best pair out of the Q subnodes generated for each subdomains is selected using the following slightly different quantity:

$$\tilde{\psi}_{Dq} = \sigma \frac{\sum_{Dq} j}{\sum_{D_l} j} \Big/ \gamma_q + (1-\sigma)\varphi_{D_q} \quad q=1,\ldots,Q \tag{17}$$

where γ_q is, for each of the subnode q, the ratio between the length of the edge along which the subdomain D_l is cut and the corresponding edge of D_q. Then once a D_q is selected the other subnode of the pair will be the complement $D_{q+1} = D_l - D_q$.

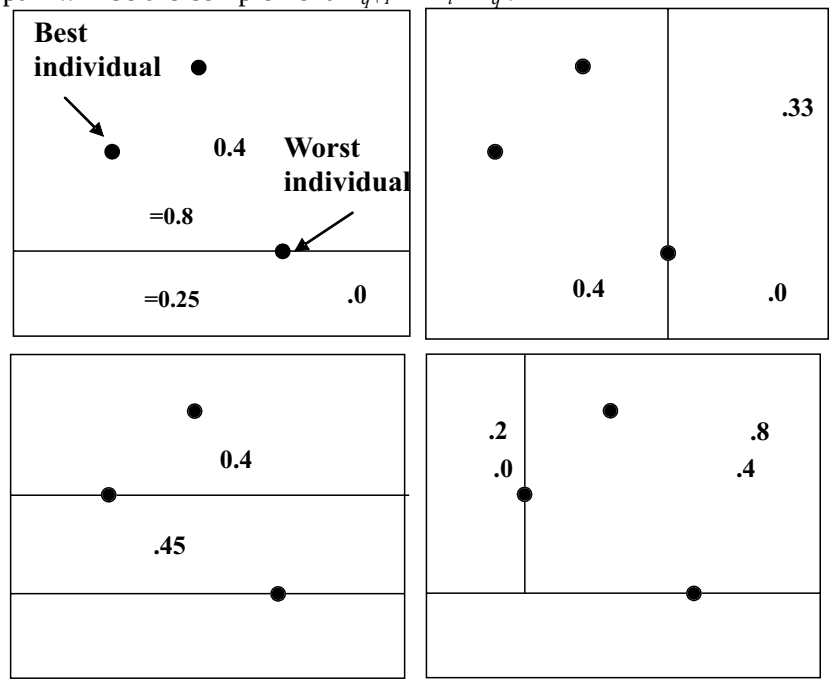

Figure 3. Sketch of the branching procedure: the first cut is performed using the worst individual, then the four resulting subregions are qualified and the one with the lowest value of $\tilde{\psi}_{Dq}$ is selected along with its complement. The procedure is then repeated using the best individual.

It should be noticed however, that, if the EP have converged in a given subdomain, nodes not containing any individual, even though they have a large volume, are unlikely to contain the solution. For a fast search, therefore, only nodes presenting high fitness and large volume are explored further.

The process is quite effective to explore the entire solution space in great detail but produces often, unnecessary re-evaluations of many regions where a local minimum has already been found. The result is a rediscovery of local minima in subdomains getting progressively smaller and smaller with a waste of computational resources. In order to avoid this phenomenon the original domain is partitioned using more than one individual. If the worst individual is useful to determine an upper bound on the objective function, converged individuals suggest where a further exploration is unnecessary. Therefore in the general scheme, all converged individuals are ranked depending on the value of their fitness function, the principal cut is then, as stated above, performed using coordinates of the worst individual, the second cut takes the worst converged individual and so on up to the best converged individual. A cartoon of the multi-partition procedure is depicted in figure 3.

Stopping Criterions

There are two combined stopping criterions: one for local convergence and one for global convergence. Both are based on some heuristics and not on any rigorous prove of global convergence. Local convergence of each subpopulation is determined by the differential improvement of the principal individual and by the migration radius. In a convex problem both should tend to zero in a neighborhood of the solution. Since each principal individual is supposed either to converge to a different minimum or not to converge (letting just the individual with highest rank in the filter to converge) a global stopping criterion for the EP is the convergence of the filter. The convergence of the filter is determined by the convergence of all the individuals, if they are not clustered, i.e. if their migration regions are not intersecting, otherwise by the convergence of the best individual. It must be noticed that when EP is used in conjunction with branching the convergence of the filter is not usually necessary since the branching takes care of the global exploration of the solution space. In this case only two cuts are performed and the best converged individual is used for the second cut.

The global convergence of the branching part is based on two ideas: the dimensions of each node and the convergence of EP in each subdomain. If a node is reduced below a given tolerance it is discarded and considered converged, therefore, if no nodes are left, the algorithm stops, on the other hand if EP have converged in all subdomains and no improvement is reported after branching, i.e. no new local minima are discovered, the algorithm stops since it is likely that all local minima have been already found and no further exploration of the solution space is required.

TEST CASES

Optimal Launch Window Problem

In an ecliptic reference frame centered into the Sun and considering the gravity action of the Sun only, the dynamic of a spacecraft is governed by the following differential system:

$$\begin{aligned} \dot{\mathbf{r}} &= \mathbf{v} \\ \dot{\mathbf{v}} &= -\frac{\mu}{r^3}\mathbf{r} \end{aligned} \tag{18}$$

where μ is the gravity constant of the Sun, $\mathbf{r}$ is the position vector of the spacecraft and $\mathbf{v}$ is its velocity vector. Now in the hypothesis of Keplerian motion taking two points in space and a fixed time of flight (TOF) T, Lambert's problem consists of finding the transfer arc from one point to the other in the given time. If this is applied to the problem of finding the optimal transfer trajectory from Earth to Mars, an infinite number of trajectories can be generated, each one

characterized by a different departure date from the Earth t_0, a different time of flight T and a different departure velocity Δv_E from the Earth and arrival velocity Δv_M at Mars. The arrival and departure velocities can be related to the cost in terms of propellant to transfer a spacecraft from the Earth to Mars, therefore the following objective function can be defined:

$$f = \Delta v_{E} + \Delta v_{M} \tag{19}$$

which must be minimized with respect to the departure time and transfer time.

If f is plotted with respect to t_0 and T the result can be seen in figures 4 and 5. If an upper limit is imposed on the maximum total Δv allowed for an interplanetary mission, the contour plot 5 shows only regions characterized by a total Δv lower than the required limit. These regions define what are generally called launch windows, i.e. intervals of possible launch dates. For the problem under study t_0 is defined in the interval [2800, 5800] expressed in Modified Julian Dates (i.e. number of days from 1st January 2000) while the TOF is defined in the interval [100,400] expressed in days. In the given domain D of launch dates and transfer times, there are at least 9 local minima but actually only one is global with a value of 5.667 km/s. However, a second minimum can be found with a slightly different value of f=5.699 km/s but for a completely different launch date.

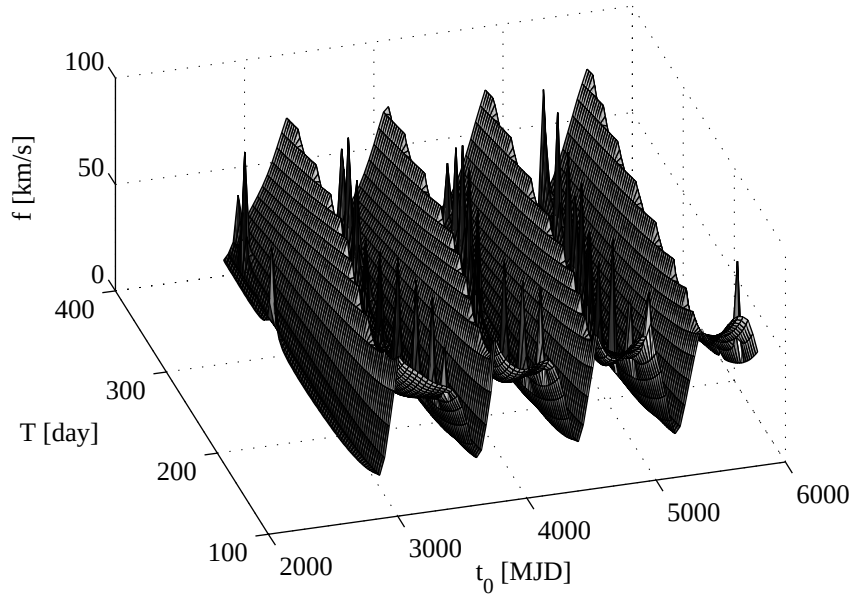

Figure 4 Three-dimensional plot of *f* for the total Δv problem.

At first only the EP algorithm is tested to verify the effectiveness and efficiency of the new operators. The problem is solved by running the EP several times, with no branching step, and checking the obtained group of minima. The stopping criterion in this case is not the complete convergence of the filter but just of the best individual. A steady population of 10 individuals has been used with a filter containing a maximum of 7 individuals. A tolerance of 1e-4 on *df* and a tolerance of 1e-4 on the migration radius have been used for the stopping criterion. Since the nature of the method is stochastic, 20 runs have been performed and the resulting number of function evaluations is the average of all 20 runs. It should be noticed that only three, out of 20, converged to the second best minimum without being captured by the basin of attraction of the best one. All the others have the global minimum in the first three positions of the filter and among them, ten have the global minimum as first value. The result of the outcome of a simple

EP step is reported in Figure 6 where all the solutions found, including those outside the filter, are represented by fat dots while actual minima are represented by circles. As can be noticed the algorithms have been able to identify three over 9 minima including the global one. However, it is clear that if the convergence of the best individual is used as a stopping criterion for the filter, exploration could be interrupted prematurely. As a consequence some minima could not be taken into account by the algorithm and among them there could be the global one especially if, as in this case, more than one minima have similar values with similar basins of attraction.

As further demonstration of the effectiveness of the algorithm the same problem has been solved using genetic algorithms, with a floating point data structure and encoding, and a pure Branching methods with a Pareto principle for box selection implemented in the code DIRECT[11]. DIRECT (Divided RECTangles) splits at each step all boxes for which the pair (volume, midpoint value) are not dominated by other such pairs. Here (v,f) is dominated by (v',f') if both $v'<v$ and $f'<f$. In particular, the box with the largest volume and the box with the best function value are never dominated and hence always split.

The Genetic Algorithms implementation used here (and freely available in the matlab toolbox GAOT[12]) uses a floating point representation of the genotype. The following genetic operators have been used on a population of 20 individuals: boundaryMutation multiNonUnifMutation nonUnifMutation unifMutation, arithXover heuristicXover simpleXover. A maximum of 500 generations are allowed and the total number of function evaluations are computed up to convergence, over 20 runs.

In the branching algorithm the number of iterations determines the stopping criterion and the total number of function evaluations. Depending on the initial domain D and therefore on the initial sampling of the solution space the number of iterations to reach a given tolerance changes dramatically. The result of the comparison among the three approaches is summarised in Table 1 where the actual global optimum is reported along with the results coming from the three optimisations, The percentage of times, over 20 runs, the algorithm was able to find the global optimum is reported next to the best value obtained for the objective function. As can be seen the systematic approach is always able to converge to the global minimum with the required accuracy. The proposed implementation of evolution algorithms performs quite well, better then both DIRECT and GAOT, thanks to the filter, to repulsion, and to the migration operator. Even though, as stated before, the global minimum is always in the filter at the end of the optimization, it is not true that the algorithm is able to recognise it all the times.

The second test includes branching and was used to verify the effectiveness of the branching criterion and to improve the exploration of the solution space.

The first run of EP spans the entire domain finding a number of minima. Some regions of the solution space result, however, unexplored since the choice of the initial population and of regenerated principal individuals is basically a random process, furthermore it might happen that even though one individual is initially in the attraction basin of a minimum, the filter rejects the individual, putting it at the bottom of the list. This happens especially when some other individuals are close to convergence. Thus, some regions result to be poorly explored because all principal individuals generated do not survive enough to converge toward a local minimum. Figure 7 reports the result of a first level of branching using the combined systematic-stochastic algorithm. The first cutting point is the worst of the individuals rejected by the filter, this ensures that the resulting branches correspond to either unexplored regions or regions containing some already found minima. Branches containing converged individuals are correctly partitioned using these individuals, and the resulting nodes with a high volume and low density as well as branches with good fitness are evaluated further. After this first branching step 8 minima over 9 have been found with good accuracy and a number of individuals are in the basins of attraction of a 9th one.

Since better minima are present in the same region the filter may converge prematurely preventing a further search for other minima in the same region. Using this technique over another 20 runs, the algorithm was always able to find the global optimum plus all the other 8 optima. Actually, it should be noticed that, due to the nature of the problem, after the first level of branching the algorithm is always able to identify the global solution. A summary of the obtained minima for the case represented in Figure 7 is reported in Table 2 where the values found by the evolution branching algorithm (EPIC) are compared to the values computed refining each solution with a SQP algorithm.

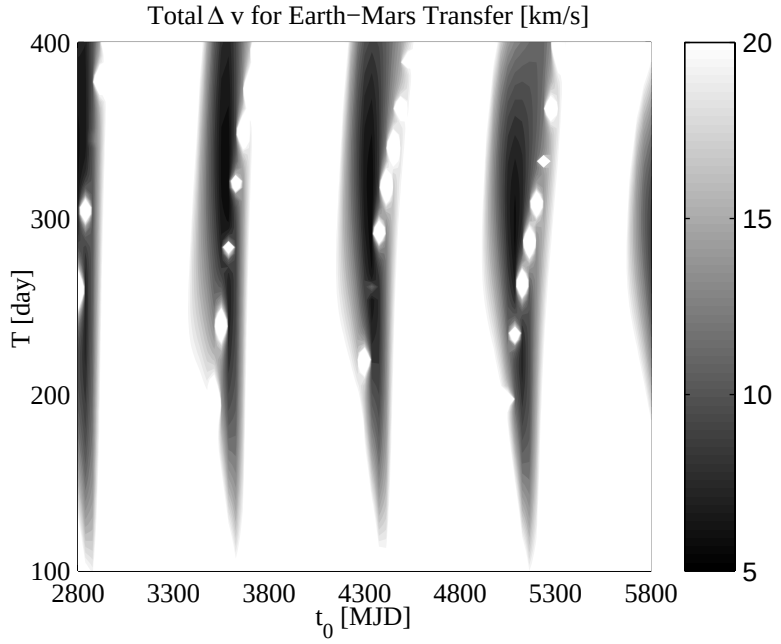

Figure 5 Contour plot of the total Δv problem.

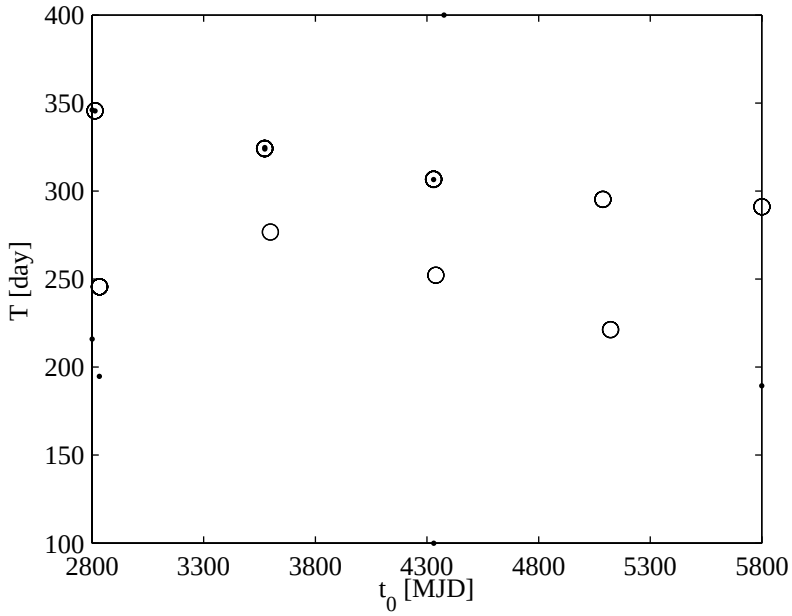

Figure 6 Result after the first evolution programming step: all individuals including those rejected by the filter are represented by fat dots while circles represent the 9 minima.

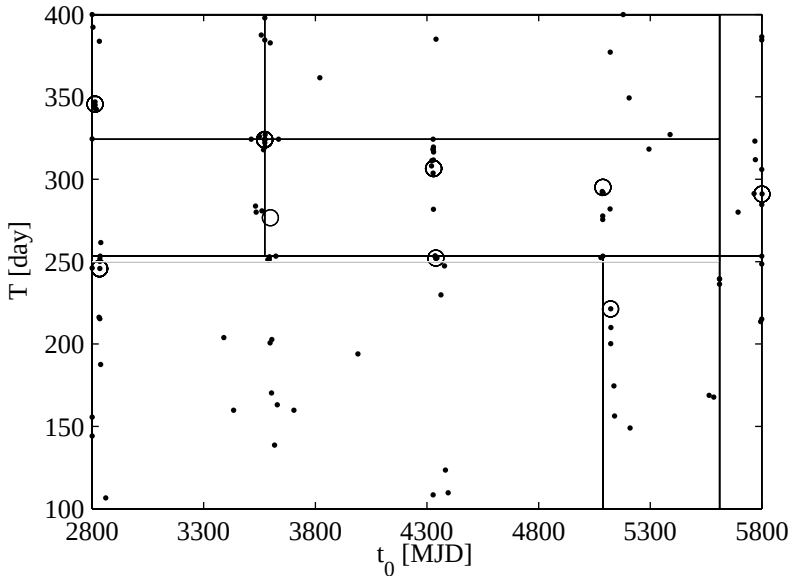

Figure 7 Result after a first level of branching with a balanced weight of 0.5.

Table.1 Pure Evolution step with convergence of the best individual

Value	Global Optimum	GAOT	DIRECT	EPIC
f (km/s)	5.6673	5.6673 (40%)	5.6674 (100%)	5.6673 (75%)
t_0 (MJD)	3573.7	3573.5	3573.9	3573.5
TOF (day)	324.05	324.11	323.85	324.34
Function Evaluations	-	1374	1183	451

Table2. Summary of minima found by EPIC

Solution	SQP	EPIC
1	(3573.7, 324.05)	(3573.7,324.06)
2	(4330.3,306.63)	(4330.3,306.65)
3	(4340.0,252.11)	(4340.5,252.15)
4	(3598.8,276.6)	(3560.5,280.58)
5	(5088.3,295.18)	(5088.3,295.18)
6	(5800.0,291.12)	(5800.0,291.12)
7	(5123.1,221.28)	(5123.1,220.95)
8	(2813.1,345.51)	(2813.1,345.46)
9	(2834.3,245.77)	(2834.3,245.79)

Optimal Earth-Mars Transfer with Electric Propulsion

In this second problem the transfer from the Earth to Mars is performed using an electric propulsion system. The departure C_3 is zero and the objective function is represented by the propellant mass which must be minimized. The engine is continuously on and the thrust direction is defined by a simple shaping law:

$$\xi = \begin{cases} \text{a}\tan(\theta + u_1) + u_2 & \text{if } \xi \le \pi/2 \\ \text{a}\tan(\theta + u_1) + u_2 - \pi & \text{otherwise} \end{cases} \qquad (20)$$
$$\lambda = abs(\text{a}\tan(20(\theta - \pi/2 + u_1))) - \pi/2$$

the dynamic of the spacecraft is therefore governed by the following system of equations:

$$\begin{aligned} \dot{\mathbf{r}} &= \mathbf{v} \\ \dot{\mathbf{v}} &= -\frac{\mu}{r^3}\mathbf{r} + \frac{T}{m}\zeta \\ \dot{m} &= -\frac{T}{g_0 I_{sp}} \end{aligned} \qquad (21)$$

where T is the maximum thrust available and it was assumed to be 0.3N, while the I_{sp} has been taken equal to 1700s and the initial mass equal to 2000kg .The control vector ζ is defined as:

$$\zeta = \Omega \begin{bmatrix} \cos\xi\sin\lambda \\ \sin\xi\sin\lambda \\ \cos\lambda \end{bmatrix} \qquad (22)$$

where Ω is the rotation matrix from the local along-track, cross-track, normal reference system attached to the spacecraft to the three dimensional Cartesian inertial reference frame (see Fig.8).

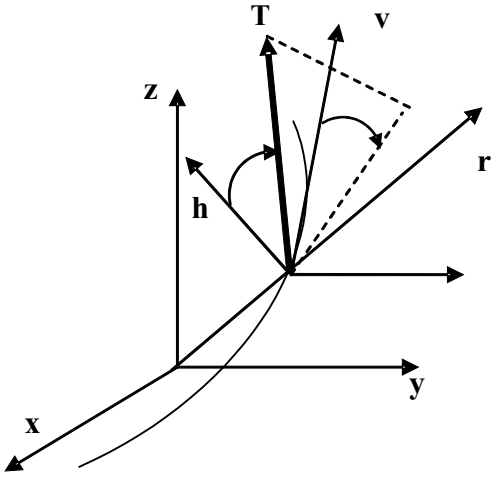

Figure 8. Local reference system

Now if each individual is encoded as a real value vector containing the departure date t_0, the transfer time P and the two parameters characterizing the shape of the control:

$$\mathbf{y} = [t_0, P, u_1, u_2]^T \qquad (23)$$

the optimization problem results to be:

$$\begin{aligned} &\min f(\mathbf{y}) = -m(t_f) \quad \text{subject to} \\ &\delta\mathbf{r} = \mathbf{r}(t_f) - \mathbf{r}_M = 0 \\ &\delta\mathbf{v} = \mathbf{v}(t_f) - \mathbf{v}_M = 0 \\ &\text{with} \quad \mathbf{y} \in D \end{aligned} \qquad (24)$$

The problem is scaled dividing the radius by 384000 (mean Earth-Moon distance), upper and lower bounds defining the domain *D* are summarized in Tab.3. The fundamental difference with respect to the previous transfer problem consists of the presence of the constraints. Furthermore, the constraints are nonlinear and do not present an explicit analytical formulation. One way to proceed is to weight the constraints in the objective function. In this particular case the weights applied have a physical meaning since the two constraints and the objective function do not have the same importance. In fact the first goal is to reach the target planet, among all solutions satisfying this constraint, only the ones with minimum arrival velocity have to be considered and among them the one with minimum propellant consumption is the desired solution. The objective function is then augmented in the following way:

$$f = (m(t_0) - m(t_f))/5000 + 2.5\delta v + \delta r \tag{25}$$

where the value 2.5 for the weight associated to velocity has been proved experimentally to give good results.

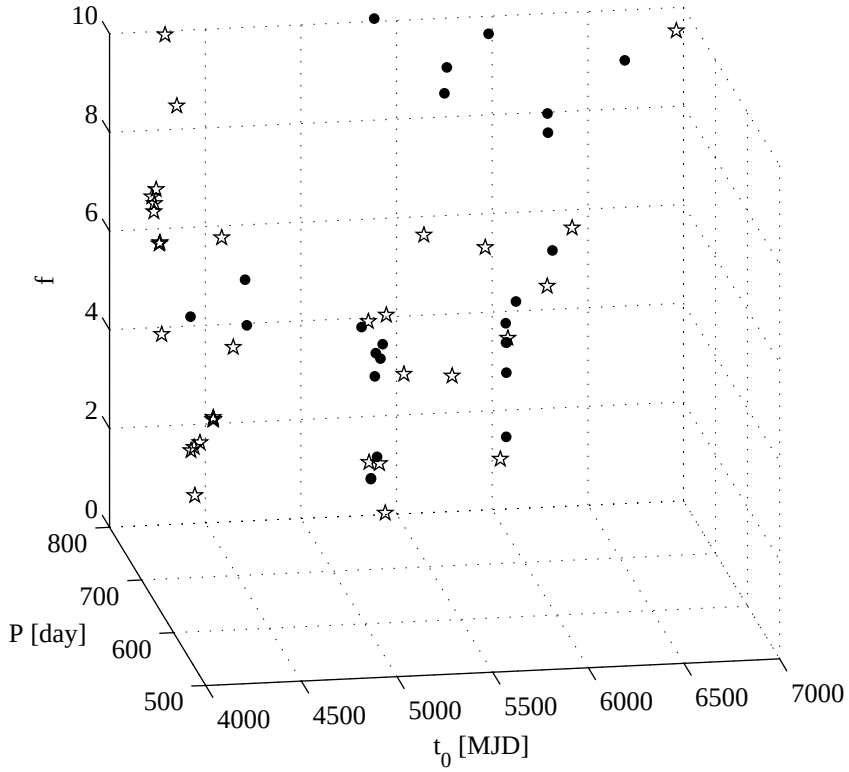

Figure 9 . Groups of solutions found by the evolution-branching algorithm

Table 3. Upper and Lower bounds for the Earth-Mars transfer problem

Value	t_0 (MJD)	P (day)	u_1	u_2
Lower bound	4000	500	$-\pi$	$-\pi$
Upper bound	7000	800	π	π

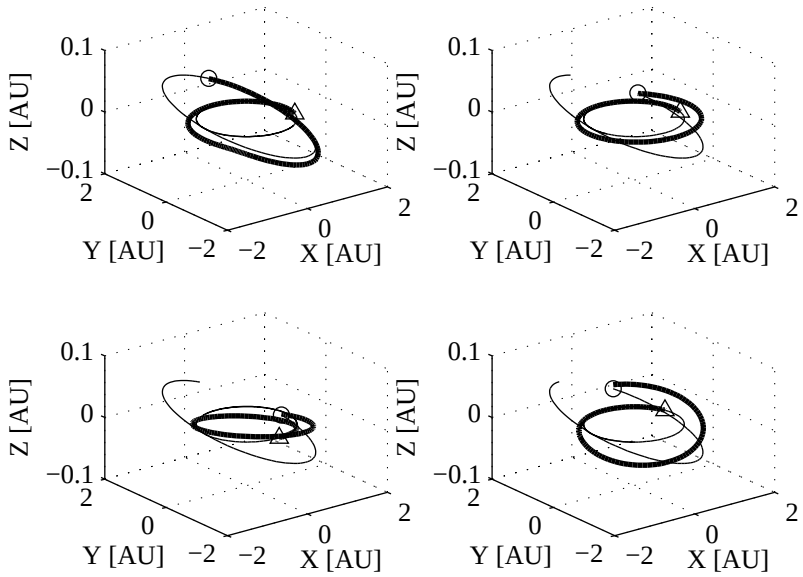

Figure 10. Examples of low-thrust transfer from the Earth, represented by a triangle, to Mars, represented by a circle.

Table 4. Evolution Branching with convergence of the best individual

Value	SQP	GAOT	DIRECT	EPIC
f	1.5977	6.2885	5.886	1.7651
m_f (kg)	926.93	1172.7	852.44	926.78
δr (km)	9.2e-6	7.6611e5	4.4679e5	7.099e4
δv (km/s)	0.55323	1.6512	1.795	0.5462
t_0 (MJD)	4261.2	5051.5	5054.9	4261.8
u_1	0.25015	0.3099	1.1997	0.24715
u_2	0.80284	-0.8427	0.0014	0.80392
P (day)	690.42	532.27	553.76	690.51
F. Eval. Single	-	7800[a]	-	3065[b]
F. Eval. Global	-	-	116860	88883

[a]The number of function evaluations is the mean value per run over 20 runs
[b]Average number of function evaluations over all the subregions explored

The combined evolution-branching algorithm is then applied to the problem with a value for the weight σ=0.9 which highly favors the exploration of unveiled regions. The branching step was iterated three times while the convergence of the best individual in the filter was used as a stopping criterion for the EP step in addition to a maximum of 50 generations. The results of two runs are reported in Fig.9 where groups or families of individuals are represented with dots and stars, for each one of the two runs respectively. It should be noticed how, in both cases, individuals tend to group in regions (or basins of attractions), corresponding to three possible launch windows, where potentially optimal solutions exist.

In order to check the accuracy of the result all the solutions obtained with EPIC have been processed with a gradient method (SQP), a comparison can be seen in Tab.4. Then the same problem has been solved also using DIRECT and GAOT and the results were reported in the same table for comparison. A maximum of 3500 iterations was used as stopping criterion for DIRECT while a maximum of 600 generations was used for GAOT (in both cases we noticed no

meaningful improvement increasing further the number of iterations or generations).

It is remarkable the number of function evaluations required by DIRECT to converge to a solution comparable with the one obtained by GAOT, which, on the other hand, was able to find a good minimum just once over 20 runs, but still 3 times worse than the best obtained by EPIC. Four different types of transfers belonging to the three launch windows found by EPIC are plotted in Fig.10 , the solution reported in Tab.4 is in the upper left corner.

CONCLUSIONS AND FUTURE WORK

In this paper a combined systematic-heuristic approach is proposed to solve trajectory design problems in which more than one solution is expected and where not just the global optimum should be obtained. The proposed combination of evolution programming and branching is suitable for problems characterized by differentiable and non-differentiable functions combining integer and real variables. The algorithm is based only on local information coming from the evolution of a limited number of individuals in subregions defined by a branching procedure. The outcome of each EP step is used to define new branches and to prune less promising ones. The particular implementation of evolution programming proposed here presents some novel operators like migration and filtering that have given quite good results, compared to Genetic Algorithms or pure branching techniques, on the problems under study yielding an independent local convergence toward several local minima. Furthermore, the particular mating procedure has demonstrated to be effective to explore widely the solution space avoiding unnecessary clustering of individuals.

Even though the obtained results must be considered preliminary, the proposed algorithm appears to be promising even for more complex space trajectory design problems. The comparison with both pure stochastic and pure systematic approaches demonstrates the effectiveness of the combination of both, in particular when the objective function is a black box. In this respect it must be said that an “ad hoc” systematic approach specifically dedicated to a certain category of trajectory design problem is expected to be more efficient.

Besides this the algorithm is quite effective as a general tool for problem with bound constraints but, at present, an extension to treat more complex constraints is under development.

Furthermore, this early implementation presents wide margins of improvement and several ideas for balanced exploration-exploitation, resource sharing, improved models for branching and migration are under study.

REFERENCES

[1] C. Bliek, P.Spellucci, L.N.Vicente, A.Neumaier, L.Granvilliers, E.Huens, P. Van Hentenryck, D. Sam-Haroud, B. Faltings. *COCONUT Deliverable D1. Algorithms for Solving Nonlinear Constrained and Optimization Problems: The State of the Art*, June 8,2001.The Coconut Project.

[2] Rauwolf G.A.,Coverstone-Carroll V.L., *Near Optimal Low-Thrust Orbit Transfers Generated by a Genetic Algorithm*. Journal of Spacecraft and Rockets, Vol.33,No.6,November-December 1996.

[3] Hartmann J.W., Coverstone V.L.,Williams S.N., *Optimal Interplanetary Spacecraft Trajectories via a Pareto Genetic Algorithm*. The Journal of the Astronautical Science, Vol. 46,No.3, July-September 1998,pp. 267-282.

[4]Vasile M., Comoretto G., Finzi A.E. *A Combination of Evolution Programming and SQP for WSB Transfer Optimisation*. AIRO2000, September 18-21 2000 Milano, Italy.

[5] Gurfil, P. and Kasdin, N. J., *Niching Genetic Algorithms-Based Characterization of Geocentric Orbits in the 3D Elliptic Restricted Three-Body Problem*, Computer Methods in Applied Mechanics and Engineering, Vol. 191, Issue 49-50, Dec. 2002, pp. 5673-5696.

[6] Debban T.J.,McConaghy T.T., Longuski, J.M., *Design and Optimization of Low-Thrust Gravity-Assist Trajetories to Selected Planets,* AIAA 2002-4729, Astrodynamic Spacialist Conference and Exhibit, 5-8 August 2002, Monterey, California USA.

[7] Vasile M. Campagnola S., Bernelli-Zazzera F. *Electric Propulsion Options for a Probe to Europa*. 2nd International Symposium on Low Thrust Trajectories (LOTUS2), Toulouse 18-20 June 2002.

[8] Michaelewicz Z. *Genetic Algorithms + Data Structures = Evolution Programs*. Springer-Verlag, Telos, 1996, Third Edition.

[9] Horn J. *The Nature of Niching: Genetic Algorithms and the Evolution of Optimal, Cooperative Populations*. Ph.D. Thesis, University of Illinois at Urbana-Champaign, 1997.

[11] Johns D.R., Perttunen C.D. and Stuckman B.E. *Lipschitzian optimization without the lipschitz constant*. J. Optim. Th. Appl., 79:157-181,1993.

[12] C.R. Houck, J.Joines, and M.Kay. *A genetic algorithm for function optimization: A Matlab implementation*. ACM Transactions on Mathematical Software, 1996.

[15] Vasile M. *A Global Optimization Algorithm for Space Trajectory Design*. ICORD 2002, 27-30 December 2002, Anna University Chennai, India.