

Global Optimisation Heuristics and Test Problems for Preliminary Spacecraft Trajectory Design

T. Vinkó and D. Izzo
email: Dario.Izzo@esa.int

Abstract—We describe a number of global optimisation problems representative of spacecraft trajectory design. Each problem is transcribed in the form of a blackbox objective function accepting, as inputs, the decision vector and returning the objective function and the constraint evaluation. The computer code is made available on line as a challenge to the community to develop performing algorithms able to solve each of the problems proposed in an efficient manner. All the problems proposed draw inspiration from real trajectory problems, ranging from Cassini to Rossetta to Messenger to possible future missions. We report the results coming from the application of standard global optimisation algorithms, with unoptimised default settings, to each one of the problems. We consider Differential Evolution, Particle Swarm Optimisation, Genetic Algorithm and Simulated Annealing. These standard implementations seem to fail to solve complex problems in a short time. We conclude the paper introducing what we call “cooperative” approaches between the different algorithms and we show how, already simple cooperation strategies, allow to improve the trajectory optimality.

INTRODUCTION

Many spacecraft trajectory design problems can be formalized as optimisation problems. Multiple gravity assist missions, with (MGA-1DSM problems) or without (MGA problems) the possibility of using deep space manouevres, are essentially global optimisation problems, which are, generally speaking, the task of finding the absolutely best value of a nonlinear function under given constraints. Good solutions to these problems can also serve as starting points for low-thrust trajectory optimisation, usually approached with local optimisation methods. Many papers have dealt with global optimisation for trajectory problems. The work of Petropoulos et al[1] identified gravity-assist trajectories to Jupiter using patched-conic techniques. Hartmann et al[2] as well as Abdelkhalik and Mortari[3] were using genetic algorithm to obtain optimal spacecraft trajectories. A paper of Izzo et al[4] introduces a deterministic search space pruning algorithm which is then applied in a combination of some heuristic global solvers and results in a very efficient method to solve MGA problems. Vasile and De Pascale[5] proposed a global search approach, which hybridizes an evolutionary-based algorithm with a systematic branching strategy for MGA problems.

It is, however, often difficult to compare the results available in literature as different techniques are repeatedly associated to different trajectory models and different bounds on the decision vector variables. This was the case, for instance, in the two Ariadna studies that the European Space Agency’s Advanced Concepts Team did in collaboration with the University of Reading [6] and with the University of Glasgow [7]. In those studies the trajectories found for some of the investigated

problems were different because of the different underlying physical models used and not because of the different set-ups of the global optimisation algorithms used. In the assessment and comparison of global optimisation techniques it is crucial to state the exact problem that is being solved releasing all the details on the model, on the algorithm used and on the variables bounds and constraints. The yearly global trajectory optimisation competition[8] (GTOC) was born to address this issue and proposes every year a trajectory optimisation problem to be solved by the world experts in the field.

The aim of this paper, which corrects and complete a previous version [9], is to propose a collection of realistic test problems as benchmarks for spacecraft trajectory global optimisation and to test on them some well-known and widely used global optimisers. All the proposed test problems are specified fully giving the bounds and units of the state variables and a comprehensive description of the constraints used. In order to provide initial results which can be used as references (as known best solutions for the proposed problems) we also report the results obtained applying our collaborative distributed global optimiser[10]. The actual objective values reached, however, should not be interpreted as (competitive) solutions to the corresponding real missions: the problems we propose and use here are not exactly the ones one would use in real missions as different bounds and constraints would most certainly be introduced as to account for particular design needs and different objective functions would be considered. These results are instead an invitation the communities –to the global optimisation community and to the aerospace engineers–, to develop, apply and compare their own algorithms on these problems. All the proposed models are coded in Matlab and C/C++ and are made available for download on our website www.esa.int/gsp/ACT/inf/op/globopt.htm in what we call the G-TOP database (Global Trajectory Optimisation Problems database) References and weblinks to the solvers used in this paper can also be found in this location. Should any research group find better solutions than the ones reported in this paper and in the web site, they are encouraged to submit them to us (act@esa.int) with a short description of the method, so that we can update the web site.

THE MODELS

In this paper we consider two trajectory models related to spacecraft equipped with high thrust engines. We refer to these problems as the MGA problem and the MGA-1DSM problem.

The first type of problem, described in details in Izzo et al.[4], represents an interplanetary trajectory of a spacecraft equipped with chemical propulsion and able to thrust only during its planetocentric phases. This simple model is useful in a number of preliminary trajectory calculations and has the advantage of resulting in a small dimensional optimisation problem that has been proven to be suitable for a pruning process having polynomial complexity both in time and in space and that results into an efficient computer implementation (GASP [4]). On the other hand, constraining the spacecraft to thrust only during the planetocentric hyperbolae is often unacceptable as it may results in trajectories that are not realistic or that use more propellant than necessary. A more complete problem is the MGA-1DSM. This represent again an interplanetary trajectory of a spacecraft equipped with chemical propulsion, able to thrust its engine once at any time between each trajectory leg. Thus the solutions to this problem are suitable to perform preliminary quantitative calculation for real space missions. This comes to the price of having to solve an optimisation problem of larger dimensions. The implementation details of this problem are the sum of a number of previously published works [5], [6], [7], [11] and thus are briefly reported. The generic form of the MGA-1DSM problem can be written as:

$$\begin{aligned} \text{find: } & \mathbf{x} \in \mathbb{R}^n \\ \text{to minimise: } & J(\mathbf{x}) \\ \text{subject to: } & \mathbf{g}(\mathbf{x}) \end{aligned}$$

where \mathbf{x} is our decision vector, J is the objective function and \mathbf{g} are non linear constraint that may come from operational considerations or from the spacecraft system design. Given a planetary sequence of N planets, the decision vector is defined by:

$$\begin{aligned} \mathbf{x} = & [t_0, V_\infty, u, v, \eta_1, T_1, \\ & r_{p2}, b_{incl2}, \eta_2, T_2, \\ & \dots, \\ & r_{pN-1}, b_{inclN-1}, \eta_{N-1}, T_{N-1}] \end{aligned}$$

As a consequence a typical MGA-1DSM problem will have dimension $d = 6 + 4(N - 2)$. In the decision vector, t_0 represent the spacecraft launch date, V_∞, u, v define the heliocentric direction of the departure hyperbolic velocity \mathbf{v}_∞ according to the formulas:

$$\begin{aligned} \theta &= 2\pi u \\ \varphi &= \arccos(2v - 1) - \pi/2 \\ \mathbf{v}_\infty/V_\infty &= \cos(\theta) \cos(\varphi) \mathbf{i} + \sin(\theta) \cos(\varphi) \mathbf{j} + \\ &+ \sin(\varphi) \mathbf{k} \end{aligned}$$

where the frame $\mathbf{i}, \mathbf{j}, \mathbf{k}$ is defined by

$$\begin{aligned} \mathbf{i} &= \mathbf{v}(t_0) / \|\mathbf{v}(t_0)\| \\ \mathbf{k} &= \mathbf{r}(t_0) \times \mathbf{v}(t_0) / \|\mathbf{r}(t_0) \times \mathbf{v}(t_0)\| \\ \mathbf{j} &= \mathbf{z} \times \mathbf{i} \end{aligned}$$

and $\mathbf{r}(t_0), \mathbf{v}(t_0)$ are the heliocentric velocity and position of the departure planet at t_0 . Once the spacecraft heliocentric

position $\mathbf{r}(t_0)$ and velocity is known $\mathbf{v}_{s/c} = \mathbf{v}(t_0) + \mathbf{v}_\infty$ its trajectory gets propagated along a keplerian orbit for the time $\eta_1 T_1$ and from the arrival point a Lambert problem [12] is solved that brings the spacecraft position to match that of the second planet in sequence in the time $(1 - \eta_1)T_1$. If $N > 2$ each subsequent i -th trajectory phase will be determined by first evaluating the fly-by geometry:

$$\begin{aligned} \tilde{\mathbf{v}}_{in} &= \mathbf{v}_{in} - \mathbf{v}_{pla} \\ e &= 1 + r_{p_i} / \mu_{pla} \|\tilde{\mathbf{v}}_{in}\| \\ \delta &= 2 \arcsin 1/e \\ \mathbf{i}_x &= \tilde{\mathbf{v}}_{in} / \|\tilde{\mathbf{v}}_{in}\| \\ \mathbf{i}_y &= \mathbf{i}_x \times \mathbf{r}_{pla} / \|\mathbf{i}_x \times \mathbf{r}_{pla}\| \\ \mathbf{i}_z &= \mathbf{i}_x \times \mathbf{i}_y \\ \tilde{\mathbf{v}}_{out} / \|\tilde{\mathbf{v}}_{in}\| &= \cos \delta \mathbf{i}_x + \sin i_B \sin \delta \mathbf{i}_y + \cos i_B \sin \delta \mathbf{i}_z \\ \mathbf{v}_{out} &= \mathbf{v}_{pla} + \tilde{\mathbf{v}}_{out} \end{aligned}$$

where μ_{pla} is the planet gravitational constant. Once the spacecraft velocity \mathbf{v}_{out} is known, we propagate the spacecraft trajectory along a keplerian orbit for the time $\eta_i T_i$. From the arrival point a Lambert problem [12] is then solved to bring the spacecraft position to match that of the $i + 1$ -th planet in sequence in the time $(1 - \eta_i)T_i$.

The objective function $J(\mathbf{x})$ typically measure the propellant consumption of the spacecraft, but can also be related to other objectives such as the total mission time, the spacecraft mass, the properties of the final orbit acquired and so on. Also the non linear constraint functions $\mathbf{g}(\mathbf{x})$ depend on the problem considered and are defined case by case.

THE PROBLEMS

In this section we describe the problems we propose as benchmarks. They are all part of the G-TOP database available from the European Space Agency web page reported below. The first two are MGA problems and the others are MGA-1DSM problems. The best solutions known (the solution vectors together with the corresponding objective values) are reported as well. These were obtained using DiGMO [10], a distributed cooperative global optimisation technique which uses several population based global solvers and able to learn the best solver combination to approach complex problems. All the problems are available for download both in MATLAB and in C++ from the G-TOP web site www.esa.int/gsp/ACT/inf/op/globopt.htm.

Cassini1

This is an MGA problem that is related to the Cassini spacecraft trajectory design problem (a more complex representation of this problem is found later). The objective of this mission is to reach Saturn and to be captured by its gravity into an orbit having pericenter radius $r_p = 108950$ km, and eccentricity $e = 0.98$. The planetary fly-by sequence considered is Earth-Venus-Venus-Earth-Jupiter-Saturn (as the

one used by Cassini spacecraft). As objective function we use the total ΔV accumulated during the mission, including the launch ΔV and the various ΔV one needs to give at the planets and upon arrival to perform the final orbit injection. For the six dimensional state vector we use the bounds given in Table I. As constraints we limit the various fly-by pericenter to the values: $r_{p1} \geq 6351.8$ km, $r_{p2} \geq 6351.8$ km, $r_{p3} \geq 6778.1$ km, $r_{p4} \geq 600000$ km. The best solution known for this problem is $\mathbf{x} = [-789.8055, 158.33942, 449.38588, 54.720136, 1024.6563, 4552.7531]$, corresponding to a final objective function of 4.93 km/sec. A graphical representation of this solution is shown in Figure 1.

TABLE I
STATE VECTOR BOUNDS IN CASSINI1.

State	Variable	LB	UB	Units
x(1)	t_0	-1000	0	MJD2000
x(2)	T_1	30	400	days
x(3)	T_2	100	470	days
x(4)	T_3	30	400	days
x(5)	T_4	400	2000	days
x(6)	T_5	1000	6000	days

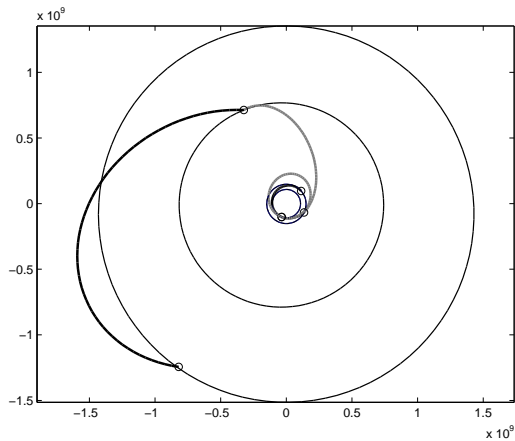


Fig. 1. Best solution known for the problem Cassini1.

GTOC1

This problem draws inspiration from the first edition of the Global Trajectory Optimisation Competition (GTOC1) [13]. More information on the yearly event can be found at www.esa.int/gsp/ACT/mad/op/GTOC/index.htm.

It is, again, an MGA problem [4] with a rather long fly-by sequence including mainly Earth and Venus. The final target is the asteroid TW229. The objective of the mission is to maximise the change in semi-major axis of the asteroid orbit following an anelastic impact of the spacecraft with the asteroid $J(\mathbf{x}) = m_f \mathbf{U} \cdot \mathbf{v}$. As constraints we limit the various fly-by pericenters to the values: $r_{p1} \geq 6351.8$ km, $r_{p2} \geq 6778.1$ km, $r_{p3} \geq 6351.8$ km, $r_{p4} \geq 6778.1$ km, $r_{p5} \geq 600000$ km, $r_{p6} \geq 70000$ km. We also consider a launcher ΔV of 2.5 km/sec, a specific impulse of $I_{sp} = 2500$ s and a spacecraft initial mass of $m_0 = 1500$ kg. For the eight dimensional state vector we use the bounds given in Table II.

TABLE II
STATE VECTOR BOUNDS IN GTOC1

State	Variable	LB	UB	Units
x(1)	t_0	3000	10000	MJD2000
x(2)	T_1	14	2000	days
x(3)	T_2	14	2000	days
x(4)	T_3	14	2000	days
x(5)	T_4	14	2000	days
x(6)	T_5	100	9000	days
x(7)	T_6	366	9000	days
x(8)	T_7	300	9000	days

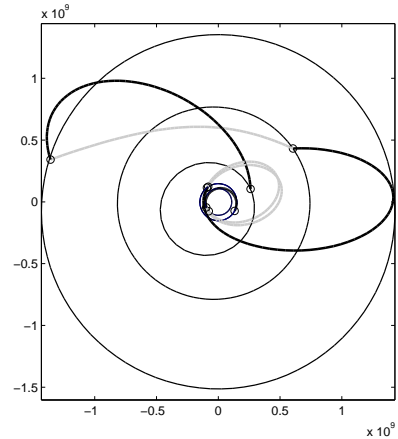


Fig. 2. Best solution known for the problem GTOC1.

The best solution known for this problem is $\mathbf{x} = [6809.476683160, 169.598512787, 1079.375156244, 56.53776494142, 1044.014046276, 3824.160968179, 1042.885114734, 3393.057868710]$, corresponding to a final objective function of 1,580,599 kg km²/sec². Figure 2 shows this solution. Note that the arc joining Saturn with the asteroid is forced to be retrograde when solving the relative Lambert problem.

SAGAS

In this trajectory problem we design what is commonly called a ΔV -EGA manoeuvre to then fly-by Jupiter and reach 50AU. The objective function considered is the overall mission length and has to be minimised. This creates an MGA-1DSM problem where two more variables need to be added to the decision vector in order to be able to evaluate the keplerian orbit reached after the last fly-by. As constraints we consider the ΔV capability of the spacecraft $\Delta V_1 + \Delta V_2 < 1.782$ and the total available $\Delta V = \Delta V_1 + \Delta V_2 + \Delta V_\infty < 6.782$ km/s. For the twelve dimension state vector we use the bounds given in Table III. Note that the bound on the departure ΔV is quite large and include a very strong minima at around 1- 4 km/sec (1:1 Earth orbit resonance) that often tricks the optimisers. Clearly, by reducing this bound (knowledge-based pruning) one can drastically help any optimiser to locate the correct global optima. As here we are interested in the algorithmic performances, we selected the bounds to create an interesting optimisation problem rather than to simplify the problem as much as possible and thus we included zones of the search space one could prune out by experience.

TABLE III
STATE VECTOR BOUNDS IN SAGAS.

State	Variable	LB	UB	Units
x(1)	t_0	7000	9100	MJD2000
x(2)	V_∞	0	7	km/sec
x(3)	u	0	1	n/a
x(4)	v	0	1	n/a
x(5)	T_1	50	2000	days
x(6)	T_2	300	2000	days
x(7)	η_1	0.01	0.9	n/a
x(8)	η_2	0.01	0.9	n/a
x(9)	r_{p1}	1.05	7	n/a
x(10)	r_{p2}	8	500	n/a
x(11)	b_{incl1}	$-\pi$	π	rad
x(12)	b_{incl2}	$-\pi$	π	rad

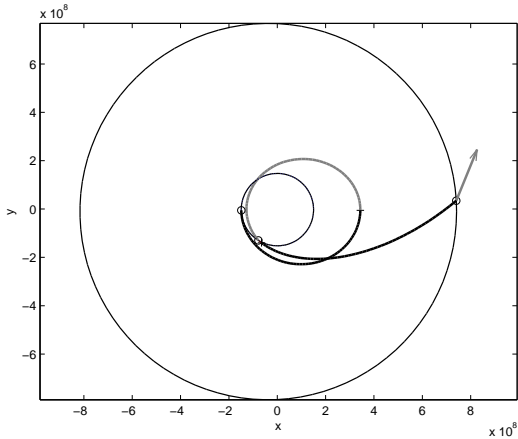


Fig. 3. Best solution known for the problem SAGAS.

The best solution known to this problem is $\mathbf{x} = [7020.49, 5.34817, 1, 0.498915, 788.763, 484.349, 0.4873, 0.01, 1.05, 10.8516, -1.57191, -0.685429]$, corresponding to a final objective function value of 18.1923 years, see Figure 3.

Cassini2

In the section named Cassini1 we presented a global optimisation problem related to the mission Cassini. In this section we consider a different model for the Cassini trajectory. This time we will allow for deep space maneuvers between each one of the planets and thus an MGA-1DSM problem. This leads to a higher dimensional problem with a much higher complexity [6], [7]. We also consider, in the objective function evaluation, a rendezvous problem rather than an orbital insertion as in Cassini1. This is the main cause for the higher objective function values reached. For the twelve dimension state vector we use the bounds given in Table IV. These are consistent with the ones used in the paper of Vasile and De Pascale[5].

No other constraints are considered for this problem. The best known solution is $\mathbf{x} = [-815.144, 3, 0.623166, 0.444834, 197.334, 425.171, 56.8856, 578.523, 2067.98, 0.01, 0.470415, 0.01, 0.0892135, 0.9, 1.05044, 1.38089, 1.18824, 76.5066, -1.57225, -2.01799, -1.52153, -1.5169]$ corresponding to an objective function value of 8.92401 km/sec, see Figure 4. In the paper by Vasile and De Pascale [5] an objective function of 9.016 km/sec is

TABLE IV
STATE VECTOR BOUNDS IN CASSINI2

State	Variable	LB	UB	Units
x(1)	t_0	-1000	0	MJD2000
x(2)	V_∞	3	5	km/sec
x(3)	u	0	1	n/a
x(4)	v	0	1	n/a
x(5)	T_1	100	400	days
x(6)	T_2	100	500	days
x(7)	T_3	30	300	days
x(8)	T_4	400	1600	days
x(9)	T_5	800	2200	days
x(10)	η_1	0.01	0.9	n/a
x(11)	η_2	0.01	0.9	n/a
x(12)	η_3	0.01	0.9	n/a
x(13)	η_4	0.01	0.9	n/a
x(14)	η_5	0.01	0.9	n/a
x(15)	\bar{r}_{p1}	1.05	6	n/a
x(16)	\bar{r}_{p2}	1.05	6	n/a
x(17)	\bar{r}_{p3}	1.15	6.5	n/a
x(18)	\bar{r}_{p4}	1.7	291	n/a
x(19)	b_{incl1}	$-\pi$	π	rad
x(20)	b_{incl2}	$-\pi$	π	rad
x(21)	b_{incl3}	$-\pi$	π	rad
x(22)	b_{incl4}	$-\pi$	π	rad

reached for a similar problem, which is very close to the best solution we found. However, details in the implementation of the problem can easily account for quite significant variation of the objective value reached (for example planetary ephemerides, different bounds on some of the variables, etc.) A recent paper of Olds, Kluever and Cupples [14] also investigates a similar problem in order to tune the parameters of a differential evolution global optimiser, they use a simpler dynamical model where the deep space manoeuvre is possible only on one preselected leg and use a different set of constraint and objective function.

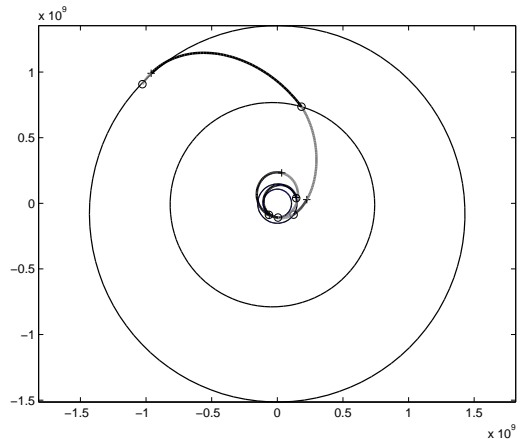


Fig. 4. Best solution known for the problem Cassini2.

Messenger

This trajectory optimisation problem represents a rendezvous mission to Mercury modelled as an MGA-1DSM problem. The selected fly-by sequence is the same used in the first part of the Messenger mission. It is well known that a significant reduction of the required ΔV is possible

if a number of resonant fly-bys follow the first Mercury encounter. Here we did not include that part of the trajectory in the optimisation problem as the dynamical model needed to represent multiple revolution solutions was not present in the code we planned to put on-line. We plan to publish the full trajectory problem description in a future work. For the eighteen dimensional global optimisation problem we consider the bounds listed in Table V.

TABLE V
STATE VECTOR BOUNDS IN MESSENGER

State	Variable	LB	UB	Units
x(1)	t_0	1000	4000	MJD2000
x(2)	V_∞	1	5	km/sec
x(3)	u	0	1	n/a
x(4)	v	0	1	n/a
x(5)	T_1	200	400	days
x(6)	T_2	30	400	days
x(7)	T_3	30	400	days
x(8)	T_4	30	400	days
x(9)	η_1	0.01	0.99	days
x(10)	η_2	0.01	0.99	n/a
x(11)	η_3	0.01	0.99	n/a
x(12)	η_4	0.01	0.99	n/a
x(13)	\bar{r}_{p1}	1.1	6	n/a
x(14)	\bar{r}_{p2}	1.1	6	n/a
x(15)	\bar{r}_{p3}	1.1	6	n/a
x(16)	b_{incl1}	$-\pi$	π	n/a
x(17)	b_{incl2}	$-\pi$	π	n/a
x(18)	b_{incl3}	$-\pi$	π	n/a

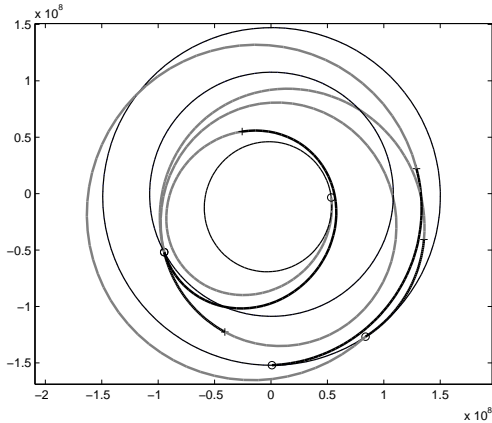


Fig. 5. Best solution known for the problem Messenger.

The best solution known for this problem is $\mathbf{x} = [2369.89 \ 1.67208 \ 0.380256 \ 0.499911 \ 400 \ 168.06 \ 224.695 \ 212.292 \ 0.237501 \ 0.0223169 \ 0.161132 \ 0.468419 \ 1.80818 \ 1.64195 \ 1.1 \ 1.29702 \ 2.80363 \ 1.57266]$ with the objective value 8.70257 km/s. This result is shown in Figure 5.

Rosetta

The problem presented in this section is a MGA-1DSM problem relative to a mission to the comet 67P/Churyumov-Gerasimenko. The fly-by sequence selected is similar to the one planned for the spacecraft Rosetta. The objective function considered is the total mission ΔV , including the launcher capabilities. The bounds used for the twenty-two dimension

decision vector are listed in Table VI. These are consistent with the ones used in Vasile and De Pascale[5].

TABLE VI
STATE VECTOR BOUNDS IN PROBLEM ROSETTA.

State	Variable	LB	UB	Units
x(1)	t_0	1460	1825	MJD2000
x(2)	V_∞	3	5	km/sec
x(3)	u	0	1	n/a
x(4)	v	0	1	n/a
x(5)	T_1	300	500	days
x(6)	T_2	150	800	days
x(7)	T_3	150	800	days
x(8)	T_4	300	800	days
x(9)	T_5	700	1850	days
x(10)	η_1	0.01	0.9	n/a
x(11)	η_2	0.01	0.9	n/a
x(12)	η_3	0.01	0.9	n/a
x(13)	η_4	0.01	0.9	n/a
x(14)	η_5	0.01	0.9	n/a
x(15)	\bar{r}_{p1}	1.05	9	n/a
x(16)	\bar{r}_{p2}	1.05	9	n/a
x(17)	\bar{r}_{p3}	1.05	9	n/a
x(18)	\bar{r}_{p4}	1.05	9	n/a
x(19)	b_{incl1}	$-\pi$	π	rad
x(20)	b_{incl2}	$-\pi$	π	rad
x(21)	b_{incl3}	$-\pi$	π	rad
x(22)	b_{incl4}	$-\pi$	π	rad

The best solution found for this problem is $\mathbf{x} = [1524.26, \ 3.83938, \ 0.262978, \ 0.779762, \ 365.574, \ 720.579, \ 262.222, \ 728.772, \ 1848.47, \ 0.194268, \ 0.225388, \ 0.273158, \ 0.671802, \ 0.407266, \ 1.61803, \ 1.06012, \ 3.33304, \ 1.08071, \ -1.32760, \ 1.85218, \ -1.44709, \ -1.97420]$, which correspond to an objective function value of $J(\mathbf{x}) = 1.4174$ km/s, see Figure 6.

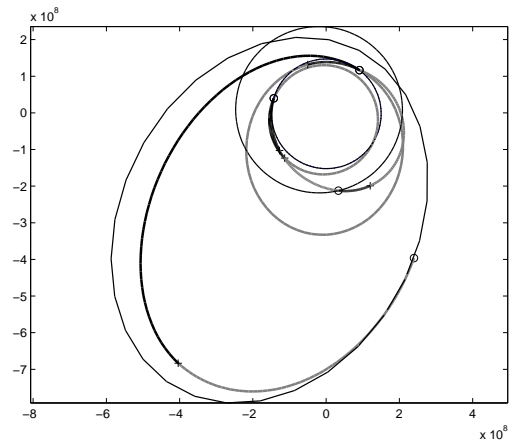


Fig. 6. Best solution known for the problem Rosetta.

THE OPTIMISERS

We here describe briefly some widely used global optimisation algorithms we will then test on the benchmark problems described to assess their efficiency. The selection of the algorithms is not exhaustive and reflects our previous experience on different global optimisation methods. The

implementation used can be downloaded from our web site www.esa.int/gsp/ACT/inf/op/globopt.htm.

- **Differential Evolution (DE):** This optimisation algorithm is based on updating each element of a set (population) of feasible solutions by using a weighted difference of two (or more) other randomly selected population elements. If the resulting element has better (lower) objective function value then this replaces the old element with which it was compared. [15]. Several variants of DE have been proposed and have been shown to be useful. In this paper we used the MATLAB implementation available from the webpage of R. Storn (one of the creators of DE, see www.icsi.berkeley.edu/~storn/code.html, but the code can also be found in our website).
- **Particle Swarm Optimization (PSO):** This is another population-based algorithm inspired by the social behaviour of bird or fish flockings [16]. In a PSO method, each element (particle) have a position and a velocity in the search space and have memory of their own best position and information on the global best position. The algorithm evolves the particles by taking the combination of the current global best and individual best solutions into account. The MATLAB implementation of the classical version from the original paper by Kennedy and Eberhart [16] was used.
- **Genetic Algorithm (GA):** The goodness of an individual in the population is measured by its fitness value (i.e. the objective function value). GA evaluates the fitness of each individual in the population and then while not converged it selects individuals to reproduce, performs crossover and mutation to make the offspring evaluates the individual fitnesses of the offspring and finally replaces the worst ranked part of the population with the offspring [17]. A simple GA implementation in MATLAB was used.
- **Simulated Annealing (SA):** Simulated Annealing [18] picks some neighbour y of a point x and compute its energy (this is like the fitness value in the above algorithms). SA moves to this new point y based on a randomly selected number which depends on the distance of the corresponding function values and on a global parameter T (temperature), that is gradually decreased during the process. Note that this algorithm is not population based. A simple MATLAB implementation was used base on the original paper by Kirkpatrick, Gelatt and Vecchi [18].

Along these standard optimisation techniques we introduce here some very simple cooperation strategies. Recent results show that collaborative usage of population based algorithms can lead to performance improvement [10].

- **COOP-1:** This algorithmic scheme is a combination of DE and PSO in a cooperative way. Namely, we allow a certain amount of iterations to each of the solvers and send the results of these "small" runs to each other. More precisely the following cycle is made: DE is called for N_1 iterations and the result population is passed to the PSO as initial value. Then PSO can take N_2 iterations and the result passed back to DE, where the whole process starts again until the allowed total number of iterations.

Typically $N_1 = N_2$ and they are much smaller than the maximum iterations allowed. Note that the velocity values used by PSO is randomly selected for each small runs of PSO.

- **COOP-2:** This algorithmic scheme is similar to the COOP-1, but GA is also involved and the three solvers are called in a randomly selected order (in COOP-1 the order is fixed).
- **COOP-3:** This is a modified version of COOP-1. Here the velocities values used by the PSO are updated also after each DE call as the difference between the old (i.e. before the actual run of DE) and the new (after the small run of DE) population.
- **COOP-4:** This algorithmic scheme is similar to the COOP-3, but GA is also used as an optimiser and these three solvers are called in a randomly selected order (in COOP-3 the order is fixed) and the PSO velocities are calculated in the same way as in COOP-3.

THE RESULTS

Here we report and discuss the experimental results obtained applying the solvers listed in the previous section. For all the solvers 20 independent runs were made and the best and the average objective values are documented. The allowed number of objective function evaluations (NFE) were limited to 200,000 for each of the problems. Note that using more (or less) function evaluations leads to different results for some of the algorithms. In order to give a better picture on the convergence history of the algorithms, we report in Figure 7 the convergence graphs for the Cassini1 problems using DE and SA. It is clear that DE converged after 10,000 function evaluations (to a local minimum) which is 5% of the allowed amount. In this case using more multistart and allowing less number of total function evaluation would result in a much better performance. Still for the Cassini1 problem SA used lot of function evaluations to converge in most of the cases. Many times it was improving its solution after using more than 100,000 function calls. In Figure 8 the convergence graphs for the Rosetta problem can be seen. Here, in many cases DE started to improve its solution after using half of the allowed function evaluations. SA also started to get better solutions after 140,000 function calls and improved its solution extensively at the end of its runs. This behaviour is strongly related with the improper (i.e. not tuned) temperature cooling scheme we used for this problem.

The following parameters were set for the different solvers:

- **DE:** population size: 20, (scale factor) $F = 0.8$, (crossover rate) $CR = 0.9$, strategy: best/2/bin.
- **PSO:** population size: 20, (inertial constant) $\omega = 0.65$, ("cognitive" and "social" constants) $\eta_1 = \eta_2 = 2$, (maximum velocity) $v_{\max} = 0.5$.
- **GA:** population size: 20, crossover rate: 0.75, mutation rate: 0.05.
- **SA:** (initial temperature) $T_0 = 100$, (the variation of the objective function that results, at T_0 in an $\exp(-1)$

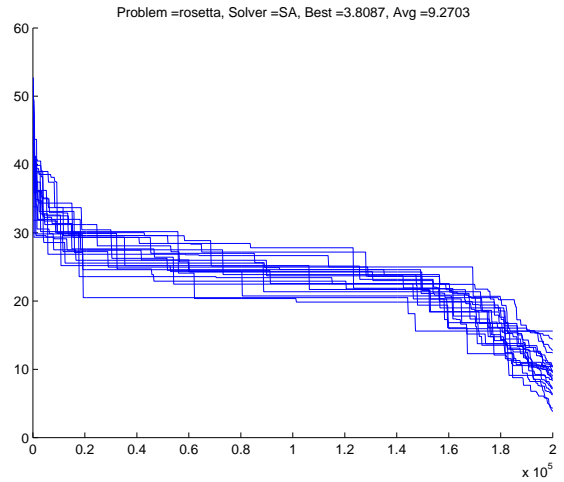
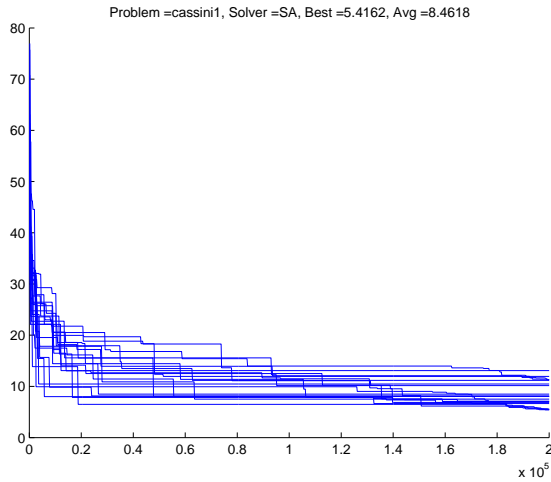
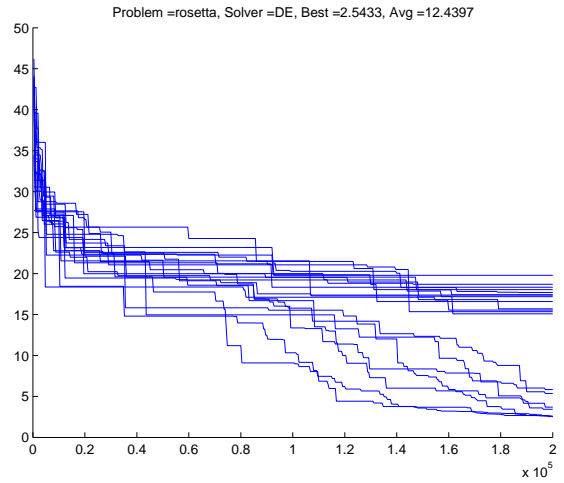
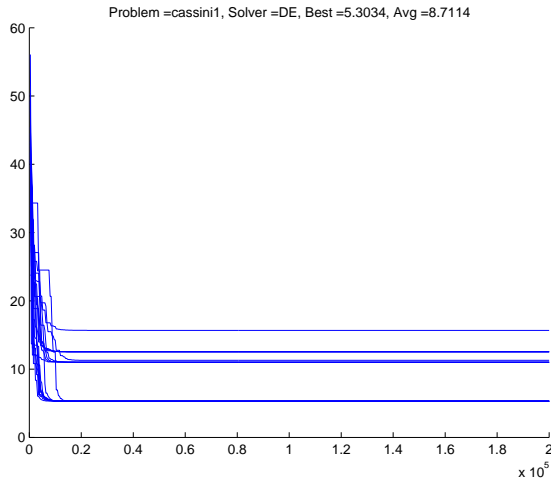


Fig. 7. Convergence graphs for Cassini1 problem with DE (left) and SA (right).

Fig. 8. Convergence graphs for Rosetta problem with DE (left) and SA (right).

acceptance rate) $jump = 10$ for all the problems, except for GTOC1 and SAGAS, where it was set up as 50,000 and 100, respectively.

- COOP: for all the versions $N_1 = N_2 (= N_3) = 20$ and the same parameters were used as the stand alone versions of DE, PSO and GA.

These values are more or less the default values commonly used with the corresponding solvers. Note that changing these parameters also leads different results. Here we do not address the problem of finding a good (or optimal) parameter setup for a particular global optimisation method in connection to trajectory optimisation problems, as this is strongly problem dependent and general statements can hardly be made.

MGA problems

First, the results for the MGA problems (Cassini1 and GTOC1) are reported in Table VII.

In case of Cassini1 none of the solvers, but one was able to locate the known best value. Note that this problem has a very strong local minima around 5.3 which introduces difficulties to the standard solvers. Three of the cooperative solvers could go down under this local minimum value and COOP-1 was

even able to find the known best solution. In average, all the SA algorithms performed very well as they were able to produce low value for all the runs, however, they stacked in local minimum all the time.

For the GTOC1 problem (note that this is a maximisation problem) the best solver is again COOP-1, however, the best solution it found is still far from the best known one. DE was the best among the standard solvers. In average, SA reached the highest function values out from 20 runs. This performance level is then followed by COOP-4 and COOP-1.

MGA-1DSM problems

For these higher dimensional problems the benchmarking results are reported in Table VIII and IX.

The SAGAS problem seems to be the most challenging to the solvers, this is related to its solution space being populated by a high number of unfeasible solutions (e.g. those with an orbit apohelium that is less than the 50AU that are the mission objective). This problem could be alleviated by a different implementation of the objective function, in particular by one able to rank also the unfeasible trajectories. As best solution among the tested methods COOP-2 reached

TABLE VII

RESULTS OBTAINED FOR THE MGA PROBLEMS. FOR ALL THE SOLVERS THE BEST AND THE AVERAGE (OUT OF 20 INDEPENDENT RUNS) OBJECTIVE VALUES ARE REPORTED.

Solver	Cassini1		GTOC1	
	best	average	best	average
DE-1	5.3034	8.7114	1,262,748	522,346
PSO	5.3946	9.0650	1,003,905	694,921
GA	5.3288	10.3103	1,018,085	450,198
SA	5.4162	8.4618	1,110,654	760,675
COOP-1	4.9307	10.9116	1,406,033	730,602
COOP-2	4.9500	9.4133	1,165,523	501,770
COOP-3	5.3034	11.1248	1,149,909	703,132
COOP-4	5.0871	9.3050	1,365,642	756,360

the lowest objective function value. All the other cooperative approaches worked well, except COOP-1 which had a rather poor performance. From the standard solvers SA reached relatively good value, although this is still very far from the best known objective value. Interestingly enough DE, PSO and GA did not work well as stand alone solvers. Much better values could be reached by using the combination of them in a cooperative way. As average, COOP-4 gave the lowest value followed by COOP-2 and COOP-3, but these values are very high numbers which mean that even these solvers were penalized during their run so they were unable to go down to the feasible (from the mission point of view) region of the search space.

For the DSM version of the Cassini mission COOP-2 and COOP-3 gave the lowest values both as best and average meaning. In this case SA did not give as good results as for the MGA problems. From the standard solvers DE found the best value (slightly worst than the ones given by the cooperative solvers). In the recent paper of Olds, Kluever and Cupples[14] DE was also confirmed to be a very efficient solver on this problem, although one should interpret this with care as the model used here is different.

TABLE VIII

RESULTS OBTAINED FOR THE MGA-1DSM PROBLEMS SAGAS AND CASSINI2. FOR ALL THE SOLVERS THE BEST AND THE AVERAGE (OUT OF 20 INDEPENDENT RUNS) OBJECTIVE VALUES ARE REPORTED.

Solver	SAGAS		Cassini2	
	best	average	best	average
DE	281.7568	892.4008	10.9008	25.3262
PSO	749.7181	1,183.0675	23.6494	28.2404
GA	579.9901	968.5306	13.1412	22.8382
SA	194.9274	891.9491	17.0156	24.0662
COOP-1	739.1383	976.6011	13.6149	20.2131
COOP-2	147.6658	746.3247	10.0519	19.8081
COOP-3	282.0148	727.4246	10.3703	19.6882
COOP-4	159.2206	691.0251	12.3563	23.5024

Considering the Messenger mission all the cooperative solvers gave very good values. The trend is, which was already noticed before, that the cooperative usage of the solvers give much better results as using them alone. This holds also for the average performance, where, in this case, COOP-1 reached the lowest value.

For the Rosetta mission COOP-1 turned out to be the best method. DE also reached a quite low value as best, however,

TABLE IX

RESULTS OBTAINED FOR THE MGA-1DSM PROBLEMS MESSENGER AND ROSETTA. FOR ALL THE SOLVERS THE BEST AND THE AVERAGE (OUT OF 20 INDEPENDENT RUNS) OBJECTIVE VALUES ARE REPORTED.

Solver	Messenger		Rosetta	
	best	average	best	average
DE	18.4613	21.0014	2.5433	12.4397
PSO	14.9187	18.3280	10.2781	14.7254
GA	12.8750	15.9063	5.9064	9.2053
SA	14.3272	17.0489	3.8087	9.2703
COOP-1	9.1982	13.4248	1.5211	4.2196
COOP-2	11.2493	14.8529	2.7542	7.0635
COOP-3	11.1867	13.3979	1.7737	4.9914
COOP-4	12.1574	15.2991	3.6472	9.1606

from its average performance it seems that this was not a trend during its run. SA gives good average values which proves again that, at least with these implementation and with these parameter setup, it is more suitable for these MGA and MGA-1DSM problems than PSO or GA.

CONCLUSIONS

We introduced six global trajectory optimisation problems in the hope they may serve as reference problems to further development of models and solvers. Test results for these problems obtained with standard global optimisation solvers were reported. We find that in every case the simple application of standard global optimisation solvers is not enough to find good solutions and some more elaborated approach is desirable. The simple scheme proposed in this paper to use collaboratively different solvers (differential evolution, particle swarm optimisation and genetic algorithm were used in here), improves the average performance of the stand alone solvers and is well suited to find good solutions to trajectory optimisation problems. A similar approach, only distributed and including more solvers, was used to present a ‘best’ solution for each one of the proposed problems.

REFERENCES

- [1] Petropoulos, A., Longuski, J., and Bonfiglio, E., “Trajectories to Jupiter via Gravity Assists from Venus, Earth, and Mars,” *Journal of Spacecraft and Rockets*, Vol. 37, 2000, pp. 776–783.
- [2] Hartmann, J., Coverstone-Carroll, V., and Williams, S., “Generation of Optimal Spacecraft Trajectories via a Pareto Genetic Algorithm,” *Journal of the Astronautical Sciences*, Vol. 46, 1998, pp. 267–282.
- [3] Abdelkhalik, O. and Mortari, D., “N-Impulse Orbit Transfer Using Genetic Algorithms,” *Journal of Spacecraft and Rockets*, Vol. 44, No. 2, 2007, pp. 456–459.
- [4] Izzo, D., Becerra, V., Myatt, D., Nasuto, S., and Bishop, J., “Search Space Pruning and Global Optimisation of Multiple Gravity Assist Spacecraft Trajectories,” *Journal of Global Optimisation*, Vol. 38, 2007, pp. 283–296.
- [5] Vasile, M. and De Pascale, P., “Preliminary Design of Multiple Gravity-Assist Trajectories,” *Journal of Spacecraft and Rockets*, Vol. 42, 2006, pp. 794–805.
- [6] Myatt, D. R., Becerra, V. M., Nasuto, S. J., and Bishop, J. M., “Advanced Global Optimisation Tools for Mission Analysis and Design,” Ariadna Final Report 03-4101a, European Space Agency, the Advanced Concepts Team, 2004, Available on line at www.esa.int/act.
- [7] Di Lizia, P. and Radice, G., “Advanced Global Optimisation Tools for Mission Analysis and Design,” Ariadna Final Report 03-4101b, European Space Agency, the Advanced Concepts Team, 2004, Available on line at www.esa.int/act.

- [8] Ongaro, F., Summerer, L., and Izzo, D., *Global Trajectory Optimization. Results of the First Competition Organised by the Advanced Concept Team (ACT) of the European Space Agency (ESA)*, Elsevier, 2007, pp. 729–816.
- [9] Vinkó, T., Izzo, D., and Bombardelli, C., “Benchmarking different global optimisation techniques for preliminary spacetrajectory design,” Paper IAC-07-A1.3.01, 58th International Astronautical Congress, Hyderabad, India, September 2007.
- [10] Vinkó, T., Izzo, D., and Pinna, F., “Learning the Best Combination of Solvers in a Distributed Global Optimisation Environment,” *Submitted for publication*, 2007.
- [11] Izzo, D., “Advances in Global optimisation For Space Trajectory Design,” Paper ISTS 2006-d-45, 25th International Symposium on Space Technology and Science, Japan, 2006.
- [12] Battin, H., *An Introduction to the Mathematics and Methods of Astrodynamics*, AIAA, 1999.
- [13] Izzo, D., “1st ACT global trajectory optimisation competition: Problem description and summary of the results,” *Acta Astronautica*, Vol. 61, 2007, pp. 731–734.
- [14] Olds, A., Kluever, C., and Cupples, M., “Interplanetary Mission Design Using Differential Evolution,” *Journal of Spacecraft and Rockets*, Vol. 44, No. 5, 2007, pp. 1060–1070.
- [15] Storn, R. and Price, K., “Differential Evolution - a Simple and Efficient Heuristic for Global Optimization over Continuous Spaces,” *Journal of Global Optimization*, Vol. 11, 1997, pp. 341–359.
- [16] Kennedy, J. and Eberhart, R. C., “Particle swarm optimization,” *Proceedings of IEEE International Conference on Neural Networks*, 1995, pp. 1942–1948.
- [17] Holland, J. H., *Adaptation in Natural and Artificial Systems*, University of Michigan Press, 1975.
- [18] Kirkpatrick, S., Gelatt, C., and Vecchi, M., “Optimization by simulated annealing,” *Science*, Vol. 220, 1983, pp. 671–680.