Engineering Optimization
Vol. 00, No. 00, August 2007, 1–19

Designing Optimal Low Thrust Gravity Assist Trajectories Using Space Pruning and a Multi-Objective Approach

Oliver Schütze[1], Max Vasile[2], Oliver Junge[3],
Michael Dellnitz[4], and Dario Izzo[5]
[1]**(corresponding author)** CINVESTAV-IPN, Computer Science Department
Mexico D.F. 07300, MEXICO
email: schuetze@cs.cinvestav.mx
[2]University of Glasgow
Department of Aerospace Engineering (http://aero.gla.ac.uk/
James Watt South Building
G12 8QQ, Glasgow,UK
email: mvasile@aero.gla.ac.uk
[3]Technische Universität München
Zentrum Mathematik / M3
Boltzmannstr. 3
D-85747 Garching, GERMANY
email: junge@ma.tum.de
[4] University of Paderborn
Institute for Industrial Mathematics
D-33098 Paderborn, Warburger Strasse 100, GERMANY
email: dellnitz@upb.de
[5]European Space Agency
Advanced Concepts Team (http://www.esa.int/act)
ESTEC, DG-PI, Keplerlaan 1
2201 AZ Noordwijk - THE NETHERLANDS
email: Dario.Izzo@esa.int
(August 2007)

In this work a multi-objective problem of finding optimal low thrust gravity assist trajectories for interplanetary and orbital transfers is addressed. For this, recently developed pruning techniques for incremental search space reduction—which will be extended for the current situation—in combination with subdivision techniques for the approximation of the entire solution set, the so-called *Pareto set*, are used. Subdivision techniques are particularly promising for the numerical treatment of these multi-objective design problems since they are characterised (amongst others) by highly disconnected feasible domains, which can easily be handled by these set oriented methods. The complexity of the novel pruning techniques is analysed, and finally the usefulness of the novel approach is demonstrated by showing some numerical results on two realistic cases.

Keywords: space mission design, low-thrust gravity assist transfers, pruning techniques, multi-objective optimisation

1 Introduction

NASAs Deep Space 1 and recently ESAs SMART-1 have shown the effectiveness of low-thrust systems as primary propulsion devices. Such new scenarios make the task of mission analysts more difficult than ever. In fact, the design of a low-thrust transfer generally requires the solution of an optimal control problem, which has no general solution in closed form. Different methods have been developed to tackle these trajectory design problems. However, all of them need to be initialised with a first guess solution. The generation of a suitable first guess turns out to be a tricky and quite time consuming task. Studies on the generation of first guess solutions for low-thrust transfers, date back to the late nineties with the works of Coverstone et al. (4, 16), where multi-objective genetic algorithms were first used to compute first guess solutions for an indirect method. The derivation of approximating analytical solutions was addressed in the works of Markopoulos (10), Bishop and Azimov (1, 2). Inspired by the work of Tanguay (20), Petropoulos and Longuski (15) proposed a shape-based approach, which represents the trajectory (connecting two points in space) with a particular parameterised analytical curve (or shape) and computes the control thrust necessary to satisfy the dynamics. Although the resulting trajectory is not the actual solution of an optimal control problem, by tuning the shaping parameters it is possible to generate solutions, which are sufficiently good to initialise a more fine optimisation process. More precisely, in the work by Petropoulos, a thrust arc is represented by an analytical curve, known as exponential sinusoid, which consists of a five parameter shape in polar coordinates. This shape is suitable for the approximation of planar motion, and the reduced number of shaping parameters does not allow to satisfy all the possible boundary conditions on position, velocity, time of flight and magnitude of the control acceleration; for 3D problems the propellant consumption for out of plane motion is only estimated. By implementing the exponential sinusoid trajectory model in the software code STOUR, Petropoulos and Longuski extended their systematic search for optimal ballistic MGA transfers to the global solution of Low-Thrust Gravity Assist (LTGA) transfers (11, 15, 13). Recently, it has been shown that whenever a Multiple Gravity Assist (MGA) optimisation problem is characterised by a simple Δv-matching for the swing-bys and no deep-space manoeuvres are present, there exists a polynomial-time algorithm (with small exponent) that provides an efficient solution to the problem (13). Namely a branch and prune technique exists, the complexity of which is quartic with respect to dimensionality, i.e. in the number of swing-bys, and cubic in the resolution of the discretisation of the time variable.
Space pruning techniques are incremental algorithms which allow pruning out infeasible regions from a given domain (or space), which is typically huge compared to the feasible set, measured by its volume. Pruning techniques have been used for decades as part of various deterministic algorithms (14), from branch and prune algorithms applicable to general black-box problems to algorithms built on specific problems, such as the one in (13). Pruning heuristics simply select from the domain D, which is often neatly defined by box constraints, a collection C of disconnected subsets $S_i \subset D$ of different shape and size. Thus, the success of the pruning techniques highly depends on the way they are integrated into the entire optimisation process.
The results in (13) are particularly interesting for two reasons: the authors demonstrated that, by exploiting problem characteristics, the search space for a particular model of MGA trajectories, could be pruned very efficiently; pruning the search space increased significantly the probability of finding a good local optimum with a stochastic based global optimisation method (in (13) the authors used an implementation of Differential Evolution). Since the pruning process was particularly efficient, the overhead in the computation cost due to the application of the pruning plus the global search was largely compensated by the increase in the reliability of the search process. Here it is understood that a search process is reliable if a given solution can be found with high probability over a number of times that the process is run.
If Multi-LTGA (MLTGA) trajectories are considered it would be desirable to have an equivalent polynomial-time algorithm. In this paper an incremental pruning algorithm is presented for the solution of the MLTGA problem and an analysis of its computational complexity is given. As in (13), it will demonstrated here that if a particular model for low-thrust arcs and for gravity assist manoeuvres is used, then an algorithm exists (which sill be called LTGASP in the following) that scales quadratically in space and quintic in time with respect to the number of gravity assist

manoeuvres considered. Further, one possible way to use the output set C to tackle the multi-objective optimisation problem (MOP) under consideration (i.e., minimisation of flight time and fuel consumption) will be shown. To be more precise, multilevel subdivision techniques will be used, which are state of the art for the numerical treatment of moderate dimensional and continuous MOPs and which can cope with disconnected domains. The resulting process—i.e., pruning and subdivision—is suitable to solve such MLTGA problems efficiently and is competitive with other existing methods.
In (13) the pruning algorithm was devised to address specifically MGA problems with a particular structure. The algorithm is therefore problem dependent and fully exploits the characteristics of the problem.
In this paper the heuristics proposed in (13) to tackle MLTGA problems will be extended. Furthermore in (13) the pruning of the solution space was intended as a way to improve the search of a stochastic-based global optimiser for single objective problems. In this paper, instead, the pruned space is used to improve the search of a multi-objective algorithm and in particular it will be shown how to improve the identification of the global Pareto front for an MLTGA problem.
The remainder of this paper is organised as follows: in Section 2 the background required for understanding the work is stated. Section 3 proposes an incremental pruning algorithm for MLTGA problems and its complexity is analysed in Section 4. Section 5 deals with the numerical treatment of the multi-objective trajectory design problem, in Section 6 some numerical results are presented, and finally conclusions are drawn in Section 7.

2 Background

In this section the required background for the understanding of the sequel is stated: the concept of multi-objective optimisation is introduced, the trajectory design problem is stated, and finally the subdivision techniques are described which will be used to attack the resulting problems.

2.1 *Multi-Objective Optimisation*

In a variety of applications in industry and finance a problem arises that several objective functions have to be optimised concurrently. One important feature of these problems is that the different objectives typically contradict each other and therefore certainly do not have identical optima. Thus, the question arises how to approximate one or several particular 'optimal compromises' or how to compute the entire set of optimal compromises – the *Pareto set* – of this *multi-objective optimisation problem* (MOP). For the solution of both problems there already exists a huge variety of efficient algorithms (see e.g. (12), (5) and references therein).
Mathematically speaking, an MOP can be stated in its general form as follows:

$$\min_{x\in\mathcal{S}}\{F(x)\}, \quad \mathcal{S}=\{x\in\mathbb{R}^n : h(x)=0,\ g(x)\le 0\},$$

where F is defined as the vector of the objectives, i.e.

$$F:\mathbb{R}^n\to\mathbb{R}^k, \quad F(x)=(f_1(x),\dots f_k(x)),$$

with $f_1,\dots,f_k:\mathbb{R}^n\to\mathbb{R}$, $h:\mathbb{R}^n\to\mathbb{R}^m$, $m\le n$, and $g:\mathbb{R}^n\to\mathbb{R}^q$. A vector $v\in\mathbb{R}^k$ is said to be *dominated* by a vector $w\in\mathbb{R}^k$ if $w_i\le v_i$ for all $i\in\{1,\dots,k\}$ and $v\ne w$. A vector v is called *nondominated* with respect to a set P, if none of the vectors $p\in P$ dominate v.
A point $x\in\mathcal{S}$ is called optimal or *Pareto optimal*, if $F(x)$ is not dominated by any vector $F(y), y\in\mathcal{S}$. The solution set—the so-called *Pareto set*—consists typically not of finitely many points as for scalar optimisation problems, but forms a $(k-1)$-dimensional object. The image of the Pareto set is called the *Pareto front*.

2.2 *The Exponential Sinusoid*

It is here proposed to use a particular model for multiple gravity assist low-thrust trajectories (MLTGA). Low-thrust arcs are modeled through a shaping approach based on the exponential sinusoid proposed by Petropoulos et al.(15). The spacecraft is assumed to be moving in a plane subject to the gravity attraction of the Sun and to the control acceleration $\mathbf{F} = [F\cos\alpha, F\sin\alpha]^T$ of a low-thrust propulsion engine. The dynamic equations governing the motion of the spacecraft can be written in polar coordinates as follows:

$$\begin{aligned} \ddot{r} - r\dot{\theta}^2 + \frac{\mu}{r^2} &= F\sin\alpha \\ \frac{1}{r}\frac{d}{dt}(r^2\dot{\theta}) &= F\cos\alpha \end{aligned}$$

where α is the thrust steering angle measured clockwise from the axis perpendicular to r in the direction of motion. For this particular dynamics Petropoulos proposed to use the following shaping function for the radius as a function of the polar angle θ.

$$r = k_0 e^{k_1 \sin(k_2\theta+\phi)}$$

Then, if the thrust vector is aligned with the velocity vector, the flight path angle γ and the thrust steering angle α are equal. If $\gamma = \alpha$, the thrust history and the polar angle history are uniquely determined and the control acceleration is given by:

$$F = \frac{\mu}{r^2}\frac{\tan\gamma}{2\cos\gamma}\left[\frac{1}{\tan^2\gamma + k_1k_2^2s + 1} - \frac{k_2^2(1-2k_1s)}{(\tan^2\gamma + k_1k_2^2s+1)^2}\right] \tag{2.1}$$

with the time variation of the true anomaly given by:

$$\dot{\theta^2} = \left(\frac{\mu}{r^3}\right)\frac{1}{\tan^2\gamma + k_1k_2^2s + 1} \tag{2.2}$$

and the flight path angle given by:

$$\tan\gamma = k_1k_2\cos(k_2\theta+\phi) \tag{2.3}$$

with $s = \sin(k_2\theta+\phi)$. Now, by solving the following integral:

$$\Delta t = \int \frac{d\theta}{\sqrt{\left(\frac{\mu}{r^3}\right)\frac{1}{\tan^2\gamma + k_1k_2^2s+1}}} \tag{2.4}$$

one can compute the actual time of flight.
The exponential sinusoid expresses the variation of the radius as a function of the polar angle θ and depends on three shaping parameters k_0, k_1, k_2 plus a phase parameter ϕ. By fixing the initial and final radius for $\theta = 0$ and $theta = \bar{\theta}$ respectively:

$$\begin{aligned} r_1 &= k_0 e^{k_1\sin(\phi)} \\ r_2 &= k_0 e^{k_1\sin(k_2\bar{\theta}+\phi)} \end{aligned} \tag{2.5}$$

two of the three parameters can be computed as a function of the others (8).
The two position radii and the angular difference $\bar{\theta}$ between the departure and the arrival points can be computed from the ephemerides of the departure and arrival planets or other celestial bodies (in this work analytical ephemerides are used). In this case it is normally required that the transfer trajectory going from one planet to the other is flown in a given time T. This implies that the actual time of flight must be equal to the required time of flight in order to have a physical solution:

$$\Delta t - T = 0 \tag{2.6}$$

If now this time constraint is solved a third parameter can be determined and the exponential sinusoid becomes a single valued function (for more details on the solution of 2.6 please refer to (8)). In this form, given the transfer time and the two position vectors at the beginning and at the end of the transfer the velocities at the two extremal points and the thrust profile can be computed. Since only one shaping parameter is free it is not possible to optimise the value of the velocities at the boundaries plus the thrust profile but the problem is equivalent to the Lambert's problem for conic arcs.
Furthermore some analysis (15) reveals that the exponential sinusoid gives physical solutions whenever $k_1 k_2^2 < 1$. This limit will be used in the remainder of this paper to limit the values of the shaping parameter k_2.

2.3 *Gravity Assist Model for the Exponential Sinusoid*

Gravity assist manoeuvres are modeled with a linked-conic approximation: the manoeuvre is instantaneous (i.e., no variation in the position of the spacecraft) and produces a deflection of the planetocentric velocity vector, the planet is reduced to a point mass with no gravity, the deflection angle β_{swing} is a function of the mass of the planet and of the incoming velocity, such that:

$$\widetilde{\mathbf{v}}_{in}^T \widetilde{\mathbf{v}}_{out} = -\widetilde{v}_i^2 \cos\beta_{swing} \tag{2.7}$$

and

$$\beta_{swing} = 2\arccos\left(\frac{\mu_p}{\widetilde{v}_{in}^2 r_p + \mu_p}\right) \tag{2.8}$$

where μ_p is the gravity constant of the swing-by planet, $\widetilde{\mathbf{v}}_{in}$ and $\widetilde{\mathbf{v}}_{out}$ are the planetocentric incoming and outgoing velocity vectors and r_p is the radius of the pericentre of the swing-by hyperbola.
Since the value of the velocities at the boundaries is not completely free, given an incoming velocity vector it is not possible, in general, to match every possible outgoing velocity vector.
A match can be obtained by inserting a ΔV correction at the pericentre of the hyperbola of the swing-by. This model will be called powered swing-by model or powered swing-by in the following.
Modeling gravity manoeuvres through powered swing-bys has a very important property: it decouples the transfer arcs one from the other. In fact each transfer arc can be computed independently from the others once the departure and arrival times of a transfer arc are defined. Than, any pair of arcs can be matched through a Δv manoeuvre. As will be shown in the remainder of the paper, this important property of this specific trajectory model, allows to device an algorithm with polynomial complexity that can incrementally prune the search space.

2.4 *Problem Formulation*

Here the bi-objective optimisation problem is described which will be considered in the sequel. The two objectives are the propellant mass fraction and the flight time of a given trajectory.

For $N+1$ celestial bodies, a sequence of thrust legs is then assembled to all the others through a sequence of powered swing-bys. Each thrust leg is modeled through a shape-based method based on exponential sinusoids (15, 8), and the first objective is given as:

$$\begin{aligned} \text{minimise:} \quad & J(y) \\ \text{subject to:} \quad & r_p \geq r_{min} \end{aligned} \tag{2.9}$$

the complete solution vector is then defined as follows:

$$y = [t_0, T_1, k_{2,1}, n_1, ..., T_i, k_{2,i}, n_i, ..., T_N, k_{2,N}, n_N]^T \tag{2.10}$$

where $k_{2,i}$ is the i-th shaping parameter for the exponential sinusoid and n_i the number of revolutions around the Sun. The objective function J is then defined as follows:

$$J = 1 - \exp\left(-\left(\frac{\Delta V_{GA} + \Delta V_0}{g_0 I_{sp1}} + \frac{\Delta V_{LT}}{g_0 I_{sp2}}\right)\right) \tag{2.11}$$

where ΔV_{GA} is the sum of all the ΔVs (variation in velocity) required to correct every gravity assist manoeuvre, ΔV_0 is the departure manoeuvre, while ΔV_{LT} is the sum of the total ΔV of each low-thrust leg. The two specific impulses I_{sp1} and I_{sp2} are respectively for a chemical engine and for a low-thrust engine and g_0 is the gravity acceleration on the surface of the Earth.
For the tests in this paper, the values $I_{sp1} = 315$s and $I_{sp2} = 2500$s are used. Note that, the computation of a transfer arc with the exponential sinusoid does not require the time history of the mass of the spacecraft. The time history of the mass would be required to compute the actual thrust profile along the transfer arc. The integration of 2.1, instead, gives directly the Δv due to the low-thrust propulsion.
The objective function 2.11 is particularly appealing because it has values ranging from 0, best transfer, to 1, worst transfer. In addition the impulsive Δv and the low-thrust Δv are weighted in such a way that the use of low-thrust is favorite with respect to the use of the impulsive corrections. Note that, the scope of this paper is not to demonstrate the suitability of the exponential sinusoid model for the design of MLTGA trajectories but rather to demonstrate the effectiveness of the proposed pruning technique.
The overall process for the composition of an MLTGA trajectory with the exponential sinusoid model can be summarised with the following steps (see Fig. 1) :

- For each departure date t_0 and N legs with transfer times $\mathbf{T} = [T_1, ..., T_i, ..., T_N]^T$
- Compute a low-thrust arc through the exponential sinusoid model from planet i to planet $i+1$ (see the transfer from A to B as an example in Fig. 1)
- Compute the incoming heliocentric velocity vector $\mathbf{v}_{in}$ and the corresponding planetocentric velocity vector $\widetilde{\mathbf{v}}_{in}$
- Compute a low-thrust arc through the exponential sinusoid model from planet $i+1$ to planet $i+2$ (see the transfer from B to C as an example in Fig. 1)
- Compute the required heliocentric outgoing velocity vector $\mathbf{v}_{rout}$ and the corresponding required planetocentric velocity vector $\widetilde{\mathbf{v}}_{rout}$
- Compute the achievable planetocentric outgoing velocity vector $\widetilde{\mathbf{v}}_{aout}$ with pericenter radius $r_p \geq r_{min}$
- If $\widetilde{\mathbf{v}}_{aout} \neq \widetilde{\mathbf{v}}_{rout}$ compute the matching ΔV_i at the pericentre of the hyperbola
- Compute the launch impulsive manoeuvre ΔV_0
- Compute the arrival impulsive manoeuvre ΔV_N
- Compute the the low-thrust ΔV_{LT}
- Compute the sum of all ΔV's

The arrival at planet i corresponds to a time $t_i = t_{i-1} + T_i$, therefore the final time at the end of the transfer is t_N, while the corrective ΔV's are computed for all the indexes from 2 to $N-1$. Note that, although all the low-thrust arcs modeled with the exponential sinusoid are planar trajectories, the whole trajectory develops in the three dimensional space. Therefore, the ΔV

required for every plane change is obtained through an impulsive manoeuvre at the boundaries of the low-thrust arc and included in ΔV_0, ΔV_N, or in the ΔV_{GA}.

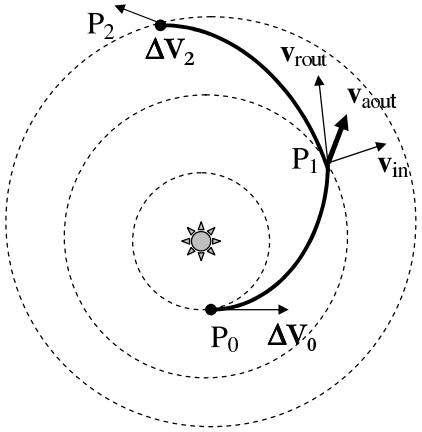


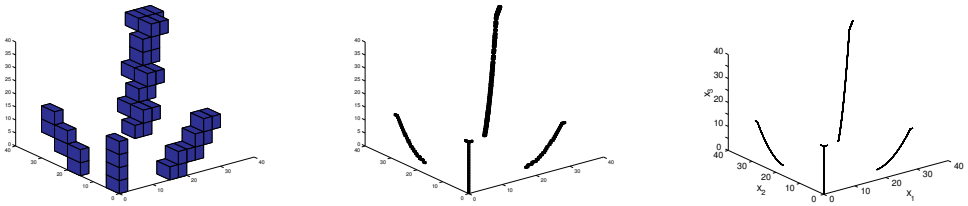
Figure 1. Composition of a whole MLTGA trajectory for the exponential sinusoid model

The second objective of the MOP which is addressed in this work is simply given by the *flight time* $t_N - t_0$ used for the selected trajectory. This objective is of great importance for the design process since the transfer can take several years (see e.g. the examples in Section 6).
Thus, the MOP under consideration reads as follows:

$$\begin{aligned} \text{minimise:} \quad & \begin{cases} J(y) \\ t_N - t_0 \end{cases} \\ \text{subject to:} \quad & r_p \geq r_{min} \end{aligned} \tag{2.12}$$

2.5 *Subdivision Techniques*

The subdivision techniques in (17), (7) have been primarily designed for MOPs without equality constraints. Algorithms of this type start with a compact subset $Q \subset \mathcal{S}$ of the parameter space, typically with one or more n-dimensional *boxes*. Each box gets subdivided into a set of smaller boxes, and according to certain conditions it is decided which box could contain a part of the Pareto set and is thus interesting for further investigation. The other, unpromising boxes are discarded from the collection. This process, i.e., subdivision and selection, is performed on the current box collection until the desired granularity of the boxes is reached. The approach is of global nature, that is, in principle capable of detecting the entire set of Pareto points, see Fig. 2 for an example. Subdivision algorithms are very effective for the numerical treatment of moderate dimensional models, but, however, may become inefficient compared other approaches like e.g. evolutionary strategies for higher-dimensional models. To combine the strength of both approaches, a combination of subivision techniques with evolutionary strategies has been proposed in (19). Though it was not required to use this hybrid method for the numerical results presented in the sequel, it is an interesting candidate for the treatment of further (higher-dimensional) models. Note that the initial box collection—as well as any further collections—(a) can in principle consist of any number of boxes and that (b) the collection does *not* have to form one connected component. Both observations are very important for the current purpose.



(a) 10 Iterations (b) 20 Iterations (c) 30 Iterations

Figure 2. Application of *DS-Subdivision* on an MOP $F : Q \subset \mathbb{R}^3 \to \mathbb{R}^2$, where Q is defined by box-constraints (see (7)).

3 LTGASP – Low Thrust Gravity Assist Space Pruning

In this section the LTGASP algorithm is proposed which is designed to efficiently detect and prune infeasible parts of the domain of a given MLTGA problem.

3.1 *The MLTGA Problem Formulation*

In order to be able to perform the pruning the numbers of revolutions in the design problem (2.12) have to be fixed. Thus, given an arbitrary but *fixed* sequence $(n_1, \ldots, n_N)$ of numbers of revolutions the following problem is considered:

$$\begin{aligned} \text{minimise:} \quad & \begin{cases} J(\tilde{y}) \\ t_N - t_0 \end{cases} \\ \text{subject to:} \quad & r_p \geq r_{min} \end{aligned} \tag{3.1}$$

where the solution vector $\tilde{y}$ is given by:

$$\tilde{y} = [t_0, T_1, T_2, \ldots, T_N, k_{2,1}, \ldots, k_{2,N}],$$

and where the parameters have the following ranges:

$$\begin{aligned} t_0 &\in \mathcal{I}_0, \\ T_i &\in \mathcal{I}_i,\ i = 1, \ldots, N, \\ k_{2,i} &\in \mathcal{I}_{k_{2,i}},\ i = 1, \ldots, N. \end{aligned}$$

Define $\mathcal{I}$ as the entire search space, i.e.

$$\mathcal{I} := \mathcal{I}_0 \times \ldots \mathcal{I}_N \times \mathcal{I}_{k_{2,1}} \times \ldots \times \mathcal{I}_{k_{2,N}}. \tag{3.2}$$

Introducing the map (analogue to (9))

$$f : x = [t_0, T_1, \ldots, T_N] \to X = [t_0, t_1, \ldots, t_N],$$

defined by the component wise relation $t_i = t_0 + \sum_{j=0}^{i} T_j$, $i = 0, \ldots, N$, and setting

$$\mathcal{I}^* := f(\mathcal{I}) \times \mathcal{I}_{k_{2,1}} \times \ldots \times \mathcal{I}_{k_{2,N}} \tag{3.3}$$

problem (3.1) can be re-formulated as:

$$\begin{aligned} \text{minimise:} \quad & \begin{cases} J(X) \\ t_N - t_0 \end{cases} \\ \text{subject to:} \quad & r_p(X) \geq r_{min}, \end{aligned} \tag{3.4}$$

which will be considered in the sequel. In the following, some notations are given which are helpful for the statement of the different pruning techniques. Every (feasible) trajectory from planet p_{i-1} to planet p_i in the current setting is determined by the parameters t_{i-1}, t_i, and $k_{2,i}$. Given these three values, denote the resulting trajectory from p_{i-1} to p_i by

$$T(t_{i-1}, t_i, k_{2,i}).$$

Further, denote by $D(\mathcal{I}_i^*)$ and $D(\mathcal{I}_{k_{2,i}})$ the discretisations of $\mathcal{I}_i^*$ and $\mathcal{I}_{k_{2,i}}$. Thus, the entire discretised search space is given by

$$D(\mathcal{I}^*) = D(\mathcal{I}_0^*) \times \ldots \times D(\mathcal{I}_N^*) \times D(\mathcal{I}_{k_{2,1}}) \times \ldots \times D(\mathcal{I}_{k_{2,N}}).$$

3.2 *The Pruning Techniques*

In the following the pruning techniques are proposed which are used for the LTGASP algorithm. A pseudocode of all algorithms can be found in Appendix 1.

Initialisation. Mark all $t_i \in D(\mathcal{I}_i^*)$, $i = 0, \ldots, n$, as valid as well as all trajectories

$$\begin{aligned} T(t_{i-1}, t_i, k_{2,i}), \quad & \forall t_{i-1} \in D(\mathcal{I}_{i-1}^*),\, t_i \in D(\mathcal{I}_i^*), \\ & k_{2,i} \in D(\mathcal{I}_{k_{2,i}}),\, i = 1, \ldots, N. \end{aligned}$$

ΔV constraining. The maximal allowable ΔV_i is the main pruning criterion of the LTGASP algorithm in phase i. It works on the sampled space $D(I_{i-1}^*) \times D(I_i^*) \times D(I_{k_{2,i}})$ and prunes out all those points corresponding to trajectories having a velocity change larger than a given budget ΔV_i^{max}.
Algorithm *1* describes the ΔV pruning for the transfer from planet p_{i-1} to planet p_i, i. e. for phase i. Denote by $\Delta V_i(T)$ the velocity change required by a given trajectory T.

Departure velocity constraining. This criterion prunes out all trajectories where the departure velocity (and thus the corresponding thrust required by the spacecraft) is larger than a given threshold.

Forward pruning. An application of the ΔV pruning in each phase typically reduces the search space volume of an MLTGA problem significantly. As a consequence many values of the arrival time t_i in phase i become nonfeasible departure times in phase $i+1$. To be more precise: if there is no feasible trajectory that arrives at a planet on a given date because they have all been pruned out according to the various criteria introduced, then there will be no departures from that planet on that date. Thus, all the corresponding points will also be pruned.
Algorithm *2* describes the forward pruning from phase i to phase $i+1$ with respect to $t_i \in D(\mathcal{I}_i^*)$.

Backward constraining. This technique is analogue to the previous one: clearly, if a departure time in phase $i+1$ becomes infeasible because of pruning, also the relative arrival date in phase i has to be pruned out.
Algorithm *3* describes the backward pruning from phase $i+1$ back to phase i with respect to $t_i \in D(\mathcal{I}_i^*)$.

Gravity assist maximum thrust constraint. The gravity assist maximum thrust constraint prunes out the trajectories having a difference between incoming velocities of trajectories in phase i (denote this velocity by $V_{end}^i(T)$ for a given trajectory T) and outgoing velocities of trajectories in phase $i+1$ (denote by $V_{start}^{i+1}(T)$) during a gravity assist larger than some threshold, A_v. This threshold has to be set separately for each gravity assist. Further, an appropriate tolerance, L_v, based on the Lipschitzian constant of the current phase plot has to be taken into account.
Algorithm *4* describes the gravity assist maximum thrust constraint pruning between phase i and phase $i+1$.

Gravity assist angular constraint. The gravity assist angular constraint prunes infeasible swing-bys from the search space on the basis of them being associated with a hyperbolic periapsis under the minimum safe distance for the given gravity assist body.
This is determined over every arrival date $\bar{t}_i \in D(\mathcal{I}_i^*)$ as follows: for all incoming trajectories $T(t_{i-1}, \bar{t}_i, k_{2,i})$ and all outgoing trajectories $T(\bar{t}_i, t_{i+1} k_{2,i+1})$ check if the corresponding swing-by is valid. In this case mark both incoming and outgoing trajectory as valid. Finally (i.e., after going through all arrival dates), all trajectories not marked as valid by this procedure will be pruned out.
Algorithm *5* describes the gravity assist angular constraint pruning between phase i and phase $i+1$.

Breaking manoeuvre constraint. As well as the departure velocity constraint, it is logical to add a constraint on the maximum breaking manoeuvre that a spacecraft can perform and prune out trajectories with an exceedingly high fuel demand.

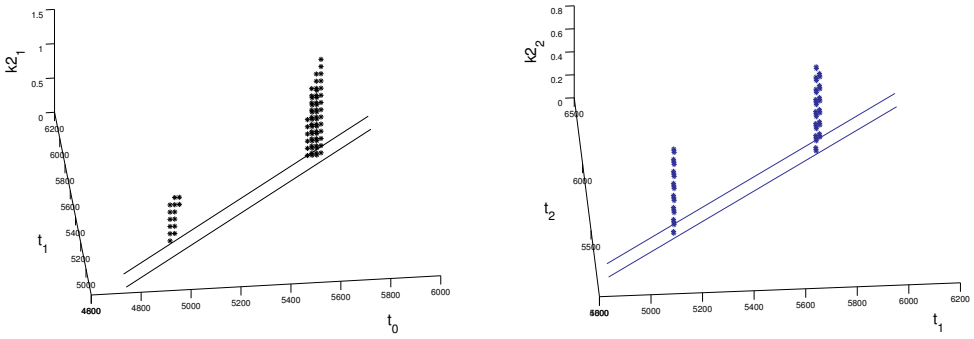
3.3 *The LTGASP Algorithm*

Having stated the different pruning techniques the complete pruning algorithm can be formulated which is done in the followng.
Given an MLTGA problem (3.4), the LTGASP algorithm for the search space reduction reads as follows:

(0) perform the initialisation process.
(1) perform the ΔV pruning, departure velocity pruning as well as the forward pruning (one 'phase shift') for phase 1.
(2) for $l = 2, \ldots, n-1$
  (a) perform the ΔV pruning for phase l.
  (b1) perform the backward pruning from phase l down to phase 1.
  (b2) perform the forward pruning from phase 1 up to phase $l+1$.
  (c) perform the gravity assist pruning for phases $l-1$ and l.
  (d) perform the angular constraint pruning for phases $l-1$ and l.
(3) perform all the pruning steps described in step (2) plus the breaking manoeuvre constraint for phase n.

Remarks 3.1*(a) This is just one possible way to combine the different pruning techniques. Note that the steps 2(a), 2(b) and 2(c) can be interchanged, and that the outcome of the resulting pruning algorithm depends on this choice. However, since the angular constraint pruning is the most time consuming technique (see the numerical results in Section 6), it is logical to apply this technique at last in each phase. Further, it is also possible e.g. to apply the backward/forward pruning after each crucial pruning criterion (such as the angular constraint pruning). This technique is typically quite effective and has—in general—a running time which is almost negligible compared to the other pruning criteria.*

(b) It has to be noted that the proposed pruning algorithm is of global nature, but does not guarantee that parts of the feasible set are not *pruned out. This is due to the fact that the algorithms work on a discretisation of the domain.*



(a) Phase 1 (b) Phase 2

Figure 3. Possible resulting candidate set of the LTGASP algorithm for a two-phase sequence (Earth – Venus – Earth) consisting of two connected components in each phase. The straight lines indicate the boundaries of the feasible subset in $t_i - t_{i+1}$ space, $i = 0, 1$, which are given by the minimal ($T_{m,i}$) and maximal ($T_{M,i}$) time of flight in each phase, i.e., $t_{i+1} \in [t_i + T_{m,i}, t_i + T_{M,i}]$.

4 Time and Space Complexity for the LTGASP Algorithm

This section determines the time and space complexity of the LTGASP algorithm. It will be shown that LTGASP scales quadratically in space and quintic in time with respect to the number of gravity assist manoeuvres considered. Since the focus is here on the order of the complexities for simplicity it can be assumed for the following analysis that the initial launch window and all phase times are the same.

4.1 *Space Complexity*

Consider a launch window, a mission phase time, and the range of the $k_2's$ discretised into l bins. Thus, for the first phase l^3 Lambert problems have to be sampled. Since the number of possible times the planet may be arrived at in phase i, $i = 1, \ldots, n$ can be assumed to be $(i+1)l$ (see (9)) the i-th phase will require an amount of $(i+1)l \cdot l \cdot l = (i+1)l^3$ Lambert function evaluations (given by the discretisations of the departure times, the time of flight and k_2). This gives the series

$$O(n) = l^3 + 2l^3 + \ldots + nl^3 = l^3 \frac{n(1+n)}{2}.$$

Therefore, the amount of space required for n phases is only of order $O(n^2)$, rather than $O(k^{2n+1})$ for full grid sampling.
Similarly, the space complexity with respect to the resolution l is of the order $O(l^3)$.

4.2 *Time Complexity*

The memory space requirement is directly proportional to the maximum number of Lambert problems that must be solved, and hence the time complexity of the sampling portion of the LTGASP algorithm must also be of the order $O(n^2)$.
For the further time complexity analysis the following assumptions are made (see above):

$$\begin{aligned} &|D(\mathcal{I}_i^*)| = (i+1)l, \quad i = 0, \ldots, n, \\ &|D(I_{k_{2,i}})| = l, \quad i = 1, \ldots, n. \end{aligned} \tag{4.1}$$

ΔV constraint complexity. The i-th step requires of the order of $i^2 l^3$ operations since by assumption (4.1) it follows that $|D(\mathcal{I}_{i-1}^*)| \cdot |D(\mathcal{I}_i^*)| \cdot |D(I_{k_{2,i}})| \approx i^2 l^3$. Thus, the time complexity applying the ΔV constraint is $O(n^3)$ with respect to the dimensionality and $O(l^3)$ with respect to the resolution.

Forward and backward pruning complexity. The forward pruning requires of the order of $i^2 l^3$ flops for one 'phase shift' (i.e., the pruning of the departure times for phase $i+1$ by analysing the data of phase i). Since for every phase i there are i such phase shifts (i.e., after the ΔV pruning of phase i and after the backward pruning), for this pruning criterion an amount of

$$\sum_{j=1}^{n} \sum_{i=1}^{j} i^2 l^3 = l^3 \sum_{j=1}^{n} \frac{j(j+1)(2j+1)}{6}$$

flops is required. Thus, the complexity of the forward pruning is $O(n^4)$ with respect to the dimensionality and $O(l^3)$ with respect to the resolution. Analogously, the complexity of the backward pruning is of the same order.

Gravity assist thrust constraint complexity. The i-th step requires of the order of $2i^2 l^3$ operations. Thus, the time complexity applying the gravity assist thrust constraint is $O(n^3)$ with respect to the dimensionality and $O(l^3)$ with respect to the resolution.

Gravity assist angular constraint complexity. The i-th step requires of the order of $2i^3 l^5$ operations. Thus, the time complexity applying the gravity assist angular constraint is $O(n^4)$ with respect to the dimensionality and $O(l^5)$ with respect to the resolution.

Overall time complexity. The overall complexity, taken from the most complex part of the algorithm (the gravity assist angular constraint), is quintic with respect to the resolution and quartic with respect to the dimensionality.

5 Attacking the MOPs

Having performed the pruning techniques resulting in a candidate set $\mathcal{C}$ which typically has a much smaller n-dimensional volume than the entire search space $\mathcal{I}$ (see Figure 3 for one example), the question arises how this information can be integrated efficiently into the optimisation process. Here one possible way is proposed to attack an MOP of the form (3.4) which involves the LTGASP algorithm presented above. Obviously, the pruning techniques have to be performed before the optimisation can start[1]. More interesting is how the set $\mathcal{C}$, which can consist of up to hundreds of different connected components of different shape and size, can be utilised for the optimisation algorithm.
The authors of this work propose to proceed in the following way:

[1]Here sequential algorithms are considered.

(1) Perform the LTGASP algorithm on the given setting. Denote the resulting candidate set by $\mathcal{C}$.
(2) Construct a *box collection* $\mathcal{R}$ starting from $\mathcal{C}$ such that all boxes are mutually non-intersecting and that $\mathcal{R}$ covers $\mathcal{C}$ 'tightly' (see discussion below).
(3) Perform the subdivision techniques described in Section 2 using collection $\mathcal{R}$ as domain.

Before some numerical results can be presented, which is done in the next section, some remarks on the approach have to be made:
A n-dimensional box B can be represented by bounds $l, u \in \mathbb{R}^n$:

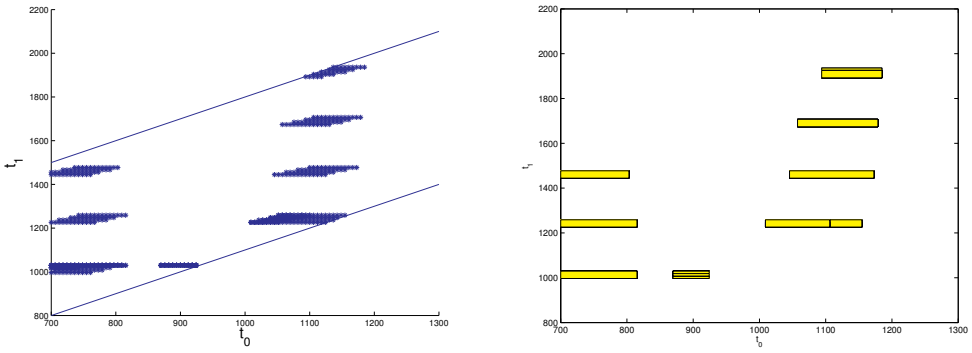
$$B = B_{l,u} = \{x \in \mathbb{R}^n : l_i \leq x_i \leq u_i \, \forall i = 1, .., n\}.$$

One way to construct a box collection $\mathcal{R}$ which covers a candidate set $\mathcal{C}$ is by looking at the connected components of the logical three-dimensional matrices A_i which correspond to the i-th phase: set $a_{jkl} = 0$ if $T(t_j, t_k, k2_l)$ is detected as not valid for the underlying sequence, where t_j is the j-th element of $|D(\mathcal{I}^*_{i-1})|$ (analogous for t_k and $k2_l$); else set $a_{jkl} = 1$. The boxes can e.g. be selected by taking the minimal and maximal coordinate values of each connected component (see Fig. 4 for an example). The corresponding n-dimensional boxes can be constructed on the basis of this sequence of three-dimensional boxes, and overlapping boxes have to be merged together.
For a given set $\mathcal{C}$ there does typically not exist one 'ideal' box collection as the following discussion shows: in order to speed up the computation, it is desired to keep the number of boxes small (since all boxes have to be evaluated). This can be done e.g. by merging 'neighboring' boxes together. However, since this increases the volume of $\mathcal{R}$ the probability of picking infeasible solutions in the run of the search procedure will increase which will in turn decrease its performance. To avoid this, smaller boxes can be considered which will in turn increase the number of boxes required to cover $\mathcal{C}$. As an example consider the set in Fig. 5 (b), which consists of 57 connected components. If components which are merely separated by one discretisation step are merged together[1] the resulting box collection consists of five boxes, which is probably the best trade off solution for this case.
For the stability of the transfer and due to the (long) resulting transfer times, up to date only few celestial bodies are involved for 'real' missions. Thus, the resulting domain of a given trajectory design problem can be considered to be low or moderate dimensional (say, ≤ 15). The authors of this work have chosen to use subdivision techniques for the approximation of the Pareto sets since the algorithms can easily cope with the disconnected domain and since they have proven their efficiency on many applications so far (e.g., (7), (18)).
On the other hand, regarding the dimensionality of the models, certainly also other approaches can in principle serve to produce satisfying results. However, note that due to the particular structure of the domain an application is in most cases not straightforward. For instance, for multi-objective evolutionary algorithms (MOEAs, see e.g., (3)), the probably most widely used class of algorithms in this field, there does ad hoc exist no suitable cross-over operator since points from different connected components of the feasible set (or from different boxes) do typically not have similar characteristics. Further, a suitable penalisation strategy is also not straightforward.

6 Numerical Results

In this section some numerical results coming from two different settings are presented. All computations have been done on an Intel Xeon 3.2 Ghz processor using the programming language MATLAB.

[1] That is, if in the logical matrix described above elements $a_{j,k,l}$ are set to 1 if $a_{j-1,k,l} = 1 = a_{j+1,k,l}$, analogous for k and l.



(a) Candidate set (b) Box collection

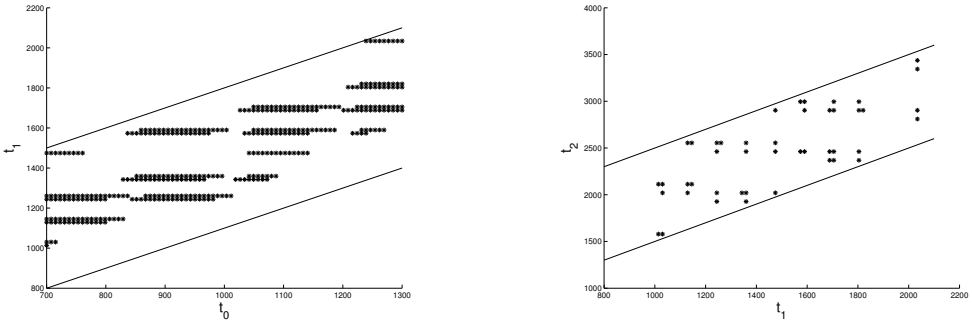
Figure 4. Relaxation of the candidate set (right) by using boxes (right), which are much easier to handle for most optimisation algorithms.

6.1 *Sequence EVMe*

First the two-phase sequence Earth – Venus – Mercury (EVMe) is under consideration. Using the parameters shown in Table 1 (see Appendix 1) the LTGASP algorithm computes the candidate set displayed in Fig. 5. Here, the left upper limit on the arrival velocity has been kept very high to allow for many possible arrival conditions at Mercury. The computation took about seven minutes (see Table 2), a time which is equivalent to evaluate approximately 3,000 different trajectories (note that there is a strong relation between the time for the pruning process and the time for a function call since for the ΔV pruning single arcs of the trajectory have to be computed). The resulting box collection consists of 47 boxes with a total volume of $5.86 \cdot 10^4$. Since the volume of the entire search space I^* (see (3.3)) is $7.18 \cdot 10^9$, LTGASP pruned out 99.9992 percent of the initial volume. Using the box collection as domain, the subdivision techniques described above obtained the Pareto front shown in Fig. 6 using a budget of $N = 50,000$ function calls.
The huge diversity in the values of the front indicates that the multi-objective approach is indeed interesting for this mission: the total time of flight varies between 200 and 1200 days which makes a difference of nearly three years, and the mass fraction varies by more than 80 %, which is certainly a huge value as well.
In order to compare the results, the subdivision techniques have been used on the same problem but without the pruning (i.e., taking $\mathcal{I}^*$ as domain). The results were not satisfying, even for much larger budgets for the number of function calls. Figure 5 shows one example for $N = 50,000$. The reason is that in the beginning of the algorithm relatively large boxes have to be evaluated where the fraction of the feasible set within these boxes are very small. Thus, it can easily happen that 'good' boxes—i.e., boxes containing a part of the Pareto set—are deleted by the algorithm. To prevent this, a huge amount of function calls has to be spent, at least in the early stages of the subdivision process. Thus, it is in this case the pruning which makes the subdivision algorithm work properly and is hence crucial for the efficiency of the proposed optimisation process.
In 7 the trajectory plot is presented for the minimum mass solution, from the best Pareto front in Fig.6. The minimum time solutions, for this specific case, corresponds to trajectories that are not physically meaningful. Trajectories of this kind are possible in the model based on the exponential sinusoid, in particular if no strict restrictions on the thrust level are imposed.

6.2 *Sequence EVEJ*

Next the three-phase sequence Earth – Venus – Earth – Jupiter (EVEJ) is considered. An application of the LTGASP algorithm using the parameter values shown in Table 4 leads to the candidate set which is shown in Fig. 8. Even in this case the arrival velocity at Jupiter is left essentially free to allow for different possible arrival conditions. For this, 12 minutes had to be



(a) Phase 1 (b) Phase 2

Figure 5. Numerical result of the LTGASP algorithm on the EVMe sequence.

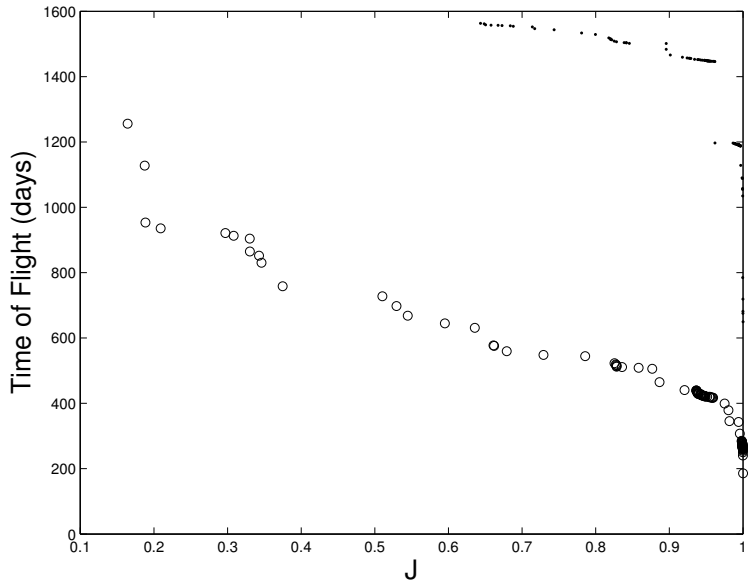


Figure 6. The circles represent a Pareto front for the EVMe sequence obtained by the subdivision algorithm with a domain $\mathcal{R}$ which is a result of the pruning algorithm LTGASP. A solution of the subdivision algorithm starting on the entire domain I^* is given by the points.

spent which corresponds to approximately 3,000 function calls. The resulting box collection $\mathcal{R}$ consists of eight boxes which have a total volume of 0.00006 percent of the volume of the entire search space I^*. Starting with this collection, the Pareto front displayed in Fig. 9 was obtained using $N = 60,000$ function evaluations. Again, the function values along the obtained front differ significantly—400 days in the transfer time and 55% in the mass fraction—leading to a large variety for the decision maker.

For a comparison to other methods a random search procedure and the state-of-the-art evolutionary algorithm NSGA-II (6) have been taken. For both approaches the entire search space I^* has been taken as domain (i.e., no pruning techniques have been applied) and a budget of

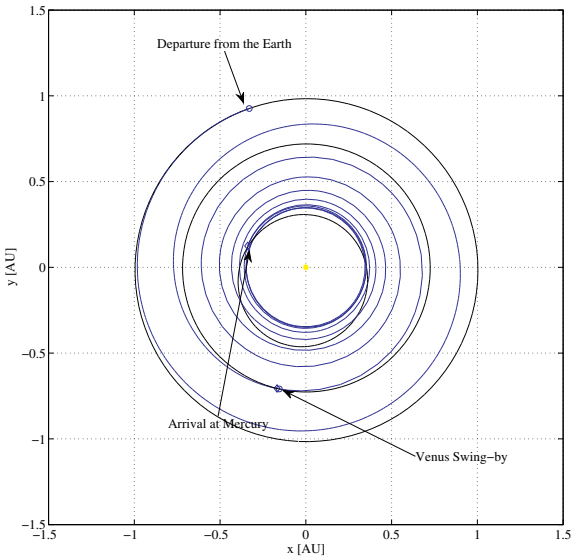
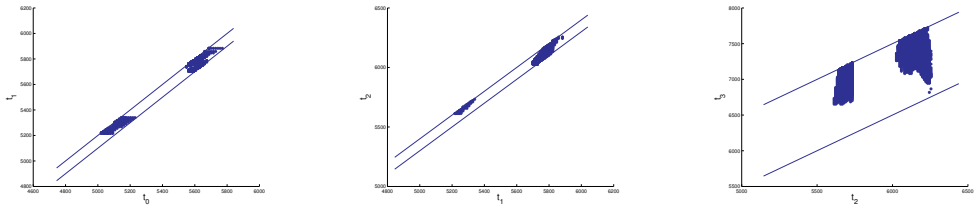


Figure 7. Minimum mass solution, for the EVMe transfer, projected onto the ecliptic plane.

$N = 100,000$ function evaluations has been given. Two representative results can be seen in Fig. 9. Apparently, none of the two results can compete with the one coming from the combination of the pruning and the subdivision. This is due to the fact that the fraction of the feasible set within I^* is tiny as the relatively small volume of $\mathcal{R}$ indicates. This might be also the reason that in this (very rare) case the random search operator outperforms NSGA-II. In such a case, a global search strategy seems to be more promising than any recombination strategy of inefficient or even useless trajectories which are obtained with a high probability, at least in the beginning of the run of the algorithm. This would change if $\mathcal{R}$ would be chosen as domain, but in that case it remains to define a suitable cross-over operator as discussed above. The result indicates that the approach proposed in this paper can cope—in contrast to existing ad hoc methods—efficiently with the given class of problems.



(a) Phase 1 (b) Phase 2 (c) Phase 3

Figure 8. Numerical result of the LTGASP algorithm on the EVEJ sequence.

In 10 the trajectory plot for the minimum mass solution (Fig.10a) is represented, from the best Pareto front in Fig.9, and the minimum time solution (Fig.10b).

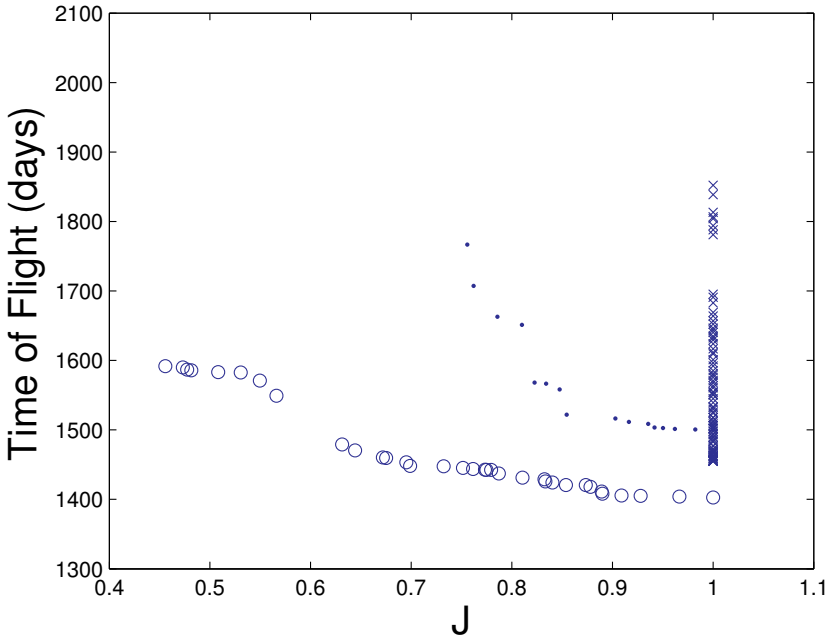


Figure 9. Different Pareto fronts for the EVEJ sequence. The crosses represent one solution obtained by NSGA-II, the points a solution coming from random search, and the circles a result obtained by pruning and subdivision.

7 Conclusions

In this work a bi-objective approach has been considered for the design of multi low-thrust gravity assist trajectories (minimisation of flight time and fuel consumption). For this, a novel way to tackle such problems, namely a combination of a space pruning technique with multilevel subdivision techniques, has been presented. In this work, pruning technique for MGA problems have been adapted and extended to the MLTGA case resulting into a deterministic algorithm called LTGASP. The LTGASP algorithm scales quadratically in space and quintic in time with respect to the number of gravity assist manoeuvres considered. The outcome set of the LTGASP algorithm can easily be manipulated such that it serves as a domain for the subdivision techniques. By doing so, the performance of this set oriented method increases significantly. The resulting optimisation process—i.e., pruning and subdivision—is suited for such design problems and is competitive to other approaches. This has been demonstrated with some numerical tests on two interplanetary transfers.

The pruning technique presented in this paper was devised specifically to address a particular MLTGA trajectory model. The accuracy of the solutions at reproducing an actual low-thrust trajectory with multiple gravity assist manoeuvre strictly depends on the accuracy of the trajectory model. Therefore, as for MGA trajectories modeled with Lambert's arcs and powered swing-bys, here it was demonstrated that for MLTGA trajectories modeled with exponential sinusoids and powered swing-bys, an efficient pruning technique exists that improves the search for Pareto optimal low-thrust transfers. Due to the shape-based transcription of the trajectory, each low-thrust arc is not optimal in the sense of the optimal control theory. Nonetheless the solutions presented in this paper are Pareto optimal with respect to the bi-objective optimisation problem and the trajectory model presented here.

8 Acknowledgment

The authors acknowledge support from the European Space Agency through the Ariadna study 05/4106. They also thank the anonymous reviewers for their valuable comments which greatly helped to them to improve the contents of this paper.

REFERENCES

[1] Bishop, R.H., Azimov, D.H., 2002. Optimal transfer between circular and hyperbolic orbits using analytical maximum thrust arcs. In *AAS/AIAA Space Flight Mechanics Meeting*, AAS Paper 02-155, San Antonio, Texas.

[2] Azimov, D.H., Bishop, R.H., 2000. New analytical solutions to the fuel-optimal orbital transfer problem using low-thrust exhaust-modulated propulsion. In *AAS/AIAA Space Flight Mechanics Meeting*, AAS Paper 00-131, Clearwater, Florida.

[3] Coello Coello, C. A., Van Veldhuizen, D. A., and Lamont, G. B., 2002. *Evolutionary Algorithms for Solving Multi-Objective Problems*. Kluwer Academic Publishers.

[4] Coverstone-Caroll, V., Hartmann, J. W., and Mason, W. M., 2000. Optimal multi-objective low-thrust spacecraft trajectories. *Computer Methods in Applied Mechanics and Engineering*, 186:387–402.

[5] Deb, K., 2001. *Multi-Objective Optimization Using Evolutionary Algorithms*. Wiley.

[6] Deb, K., Pratap, A., Agarwal, S., and Meyarivan, T., 2002. A Fast and Elitist Multiobjective Genetic Algorithm: NSGA–II. *IEEE Transactions on Evolutionary Computation*, 6(2):182–197.

[7] Dellnitz, M., Schütze, O., and Hestermeyer, T., 2005. Covering Pareto sets by multilevel subdivision techniques. *Journal of Optimization Theory and Applications*, 124:113–155.

[8] Izzo, D., 2006. Lambert's problem for exponential sinusoids. *Journal of Guidance Control and Dynamics*, 29(5):1242–1245.

[9] Izzo, D., Becerra, V. M., Myatt, D. R., Nasuto, S. J., and Bishop, J. M., 2007. Search space pruning and global optimisation of multiple gravity assist spacecraft trajectories. *Journal of Global Optimisation*, 38:283–296.

[10] Markopoulos, N., 1995. Non-keplerian manifestations of the keplerian trajectory equation, and a theory of orbital motion under continuous thrust. Paper AAS 95-217.

[11] Debban, T. J., Petropoulos, A. E., Longuski, J. M., and McConaghy, T.T., 2001. An approach to design and optimization of low-thrust trajectories with gravity-assist. In *AAS/AIAA Astrodynamics Specialists Conference*, AAS Paper 01-468, Quebec City, Canada.

[12] Miettinen, K., 1999. *Nonlinear Multiobjective Optimization*. Kluwer Academic Publishers.

[13] Myatt, D. R., Becerra, V. M., Nasuto, S. J., and Bishop, J. M., 2004. Advanced global optimisation tools for mission analysis and design. Ariadna Final Report 03-4101a, European Space Agency, the Advanced Concepts Team. Available on line at www.esa.int/act.

[14] Neumaier, A., 2004. Complete search in continuous global optimization and constraint satisfaction. *Acta Numerica 2004*, pages 271–369.

[15] Longuski, J. M., Vinh, N. X., and Petropoulos, A.E., 1999. Shape-based analytical representations of low-thrust trajectories for gravity-assist applications. In *AAS/AIAA Astrodynamics Specialists Conference*, AAS Paper 99-337, Girdwood, Alaska.

[16] Prussing, J. E., and Coverstone, V., 1998. Constant radial thrust acceleration redux. *Journal of Guidance Control and Dynamics*, 21(3):516-518.

[17] Schütze, O., 2004. *Set Oriented Methods for Global Optimization*. PhD thesis, University of Paderborn. <http://ubdata.uni-paderborn.de/ediss/17/2004/schuetze/>.

[18] Schütze, O., Jourdan, L., Legrand, T., Talbi, E.-G., and Wojkiewicz, J. L., 2007. A multi-objective approach to the design of conducting polymer composites for electromagnetic shielding. In S. Obayashi, K. Deb, C. Poloni, T. Hiroyasu, and T. Murata, editors, *Evolutionary Multi-Criterion Optimization*, Lecture Notes in Computer Science.

[19] Schütze, O., Mostaghim, S., Dellnitz, M., and Teich, J., 2003. Covering Pareto sets by multilevel evolutionary subdivision techniques. In C. M. Fonseca, P. J. Fleming, E. Zitzler, K. Deb, and L. Thiele, editors, *Evolutionary Multi-Criterion Optimization*, Lecture Notes in Computer Science.

[20] Tanguay, A. R., 1960. *Space Maneuvers, Space Trajectories.* Academic Press, New York.

9 Appendix 1: The Pruning Algorithms

This section gives the pseudocodes for all the different pruning techniques described in Section 3.2.

Algorithm *1* ΔV pruning

for all valid $t_{i-1} \in D(\mathcal{I}^*_{i-1})$ **do**
  for all valid $t_i \in D(\mathcal{I}^*_i)$ **do**
    if $t_i - t_{i-1} \in \mathcal{I}_i$ **then**
      for all $k_{2,i-1} \in D(\mathcal{I}_{k_{2,i-1}})$ **do**
        if $\Delta V_i(T(t_{i-1}, t_i, k_{2,i})) > \Delta V_i^{max}$ **then**
          mark $T(t_{i-1}, t_i, k_{2,i})$ as not valid.
        end if
      end for
    end if
  end for
end for

Algorithm *2* Forward pruning

for all valid $\bar{t}_i \in D(\mathcal{I}^*_i)$ **do**
  If $T(t_{i-1}, \bar{t}_i, k_{2,i})$ is not valid for all
  $(t_{i-1}, k_{2,i}) \in D(\mathcal{I}^*_i) \times D(\mathcal{I}_{k_{2,i}})$, mark $\bar{t}_i$ as
  not valid as well as all trajectories $T(\bar{t}_i, t_{i+1}, k_{2,i+1})$
end for

Algorithm *3* Backward pruning

for all valid $\bar{t}_i \in D(\mathcal{I}^*_i)$ **do**
  If $T(\bar{t}_i, t_{i+1}, k_{2,i+1})$ is not valid for all $(t_{i+1}, k_{2,i+1}) \in D(\mathcal{I}^*_{i+1}) \times D(\mathcal{I}_{k_{2,i+1}})$,
  mark $\bar{t}_i$ as not valid as well as all trajectories $T(t_{i-1}, \bar{t}_i, k_{2,i})$
end for

10 Appendix 2: Details on the Numerical Results

This section shows the parameter settings and the running times for the different sequences considered in Section 6.

Algorithm 4 Gravity assist maximum thrust constraint pruning

for all valid $\bar{t}_i \in D(\mathcal{I}_i^*)$ **do**
  $v_{min}^f := \min_{t_{i-1},k_{2,i}} V_{end}^i(T(t_{i-1},\bar{t}_i,k_{2,i}))$ ▷ forward
  $v_{max}^f := \max_{t_{i-1},k_{2,i}} V_{end}^i(T(t_{i-1},\bar{t}_i,k_{2,i}))$
  for all valid $t_{i+1} \in D(\mathcal{I}_{i+1}^*))$ **do**
    for all valid $k_{2,i+1} \in D(\mathcal{I}_{k_{2,i+1}}^*))$ **do**
      if $V_{start}^{i+1}(T(\bar{t}_i,t_{i+1},k_{2,i+1})) \notin [v_{min}^f - A_v - L_v, v_{max}^f + A_v + L_v]$ **then**
        mark $T(\bar{t}_i,t_{i+1},k_{2,i+1})$ as not valid.
      end if
    end for
  end for
  $v_{min}^b := \min_{t_{i+1},k_{2,i+1}} V_{start}^{i+1}(T(\bar{t}_i,t_{i+1},k_{2,i+1}))$ ▷ backward
  $v_{max}^b := \max_{t_{i+1},k_{2,i+1}} V_{start}^{i+1}(T(\bar{t}_i,t_{i+1},k_{2,i+1}))$
  for all valid $t_{i-1} \in D(\mathcal{I}_{i-1}^*))$ **do**
    for all valid $k_{2,i} \in D(\mathcal{I}_{k_{2,i}}^*))$ **do**
      if $V_{end}^i(T(t_{i-1},\bar{t}_i,k_{2,i})) \notin [v_{min}^b - A_v - L_v, v_{max}^b + A_v + L_v]$ **then**
        mark $T(t_{i-1},\bar{t}_i,k_{2,i})$ as not valid.
      end if
    end for
  end for
end for

Algorithm 5 Gravity assist angular constraint pruning

for all $\bar{t}_i \in D(I_i^*)$ **do**
  for all valid incoming trajectories $T(t_{i-1},\bar{t}_i,k_{2,i})$ **do**
    for all valid outgoing trajectories $T(\bar{t}_i,t_{i+1},k_{2,i+1})$ **do**
      if the swing-by for $T(t_{i-1},\bar{t}_i,k_{2,i})$ and $T(\bar{t}_i,t_{i+1},k_{2,i+1})$ is valid **then**
        mark $T(t_{i-1},\bar{t}_i,k_{2,i})$ as valid
        mark $T(\bar{t}_i,t_{i+1},k_{2,i+1})$ as valid
      end if
    end for
  end for
end for
Invalidate all trajectories not marked as valid

Table 1. Parameter settings for the pruning of the EVMe sequence. The shaping parameters have been fixed to $k_{2,1} = 0.27$ and $k_{2,2} = 0.007$

sequence	:	Earth – Venus – Mercury
launch window	:	$[700, 1300]$ (days after 01.01.2000)
time of flight phase 1	:	$[100, 800]$ (days)
time of flight phase 2	:	$[500, 1500]$ (days)
$\|D(\mathcal{I}_0^*)\|$	:	80
$\|D(\mathcal{I}_1^*)\|$	:	80
$\|D(\mathcal{I}_2^*)\|$	:	100
ΔV_1^{max}	:	5 (km/s)
ΔV_2^{max}	:	5 (km/s)
max. departure velocity	:	10 (km/s)
max. terminal velocity	:	30 (km/s)
r_{min} Venus	:	6750 (km)

Table 2. Running times for sequence EVMe (see Section 6.1).

ΔV pruning 1. phase	:	185.64 sec.
ΔV pruning 2. phase	:	167.55 sec.
backward/forward pruning	:	0.03 sec.
max. thrust pruning	:	0.29 sec.
ang. constr. pruning	:	26.05 sec.
2nd backward/forward pruning	:	0.02 sec.
final backw./forw. pruning	:	0.17 sec.
total running time	:	379.75 sec.

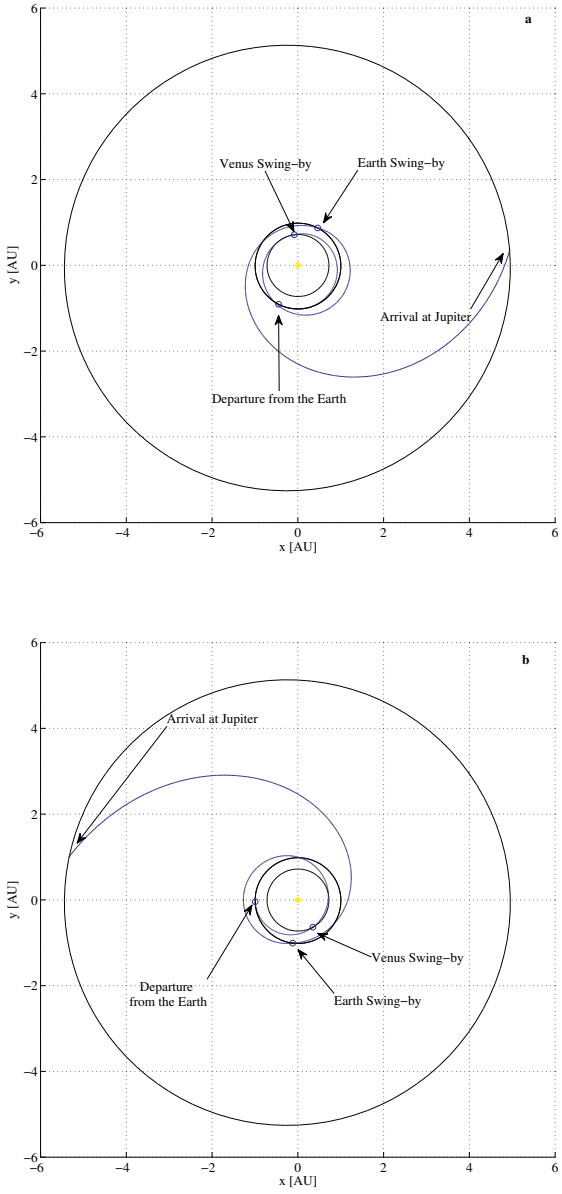


Figure 10. a) minimum mass solution, for the EVEJ transfer, projected onto the ecliptic plane, b) minimum time solution, for the EVEJ transfer, projected onto the ecliptic plane.

Table 3. Parameter settings for the pruning of the EVEJ sequence.

sequence	:	Earth – Venus – Earth – Jupiter
launch window	:	$[3745, 6840]$ (days after 01.01.2000)
time of flight phase 1	:	$[100, 200]$ (days)
time of flight phase 2	:	$[300, 400]$ (days)
time of flight phase 3	:	$[1000, 2000]$ (days)
$\|D(\mathcal{I}_0^*)\|$	:	80
$\|D(\mathcal{I}_1^*)\|$	:	80
$\|D(\mathcal{I}_2^*)\|$	:	100
$\|D(\mathcal{I}_3^*)\|$	:	120
range of $k_{2,i}, i = 1, 2, 3$	:	$[0.01, 2]$
$\|D(I_{k_{2,1}})\|$	:	20
$\|D(I_{k_{2,2}})\|$	:	20
$\|D(I_{k_{2,3}})\|$	:	20
ΔV_1^{max}	:	5 (km/s)
ΔV_2^{max}	:	5 (km/s)
ΔV_3^{max}	:	5 (km/s)
max. departure velocity	:	10 (km/s)
max. terminal velocity	:	30 (km/s)
r_{min} Venus	:	6750 (km)
r_{min} Earth	:	6750 (km)

Table 4. Running times for sequence EVEJ (see Section 6.2).

ΔV pruning 1. phase	:	66.42 sec.
ΔV pruning 2. phase	:	38.88 sec.
backward/forward pruning	:	0.06 sec.
max. thrust pruning	:	1.48 sec.
ang. constr. pruning	:	374.20 sec.
backward/forward pruning	:	0.06 sec.
ΔV pruning 3. phase	:	180.63 sec.
backward/forward pruning	:	0.09 sec.
max. thrust pruning	:	1.26 sec.
ang. constr. pruning	:	97.39 sec.
backward/forward pruning	:	0.08 sec.
final backw./forw. pruning	:	1.16 sec.
total running time	:	761.72 sec.