Advanced Global Optimisation for Mission Analysis and Design

Authors: D.R. Myatt, V.M. Becerra, S.J. Nasuto, J.M. Bishop
Contractor: Department of Cybernetics, The University of Reading, UK.

Technical officer: Dario Izzo, Advanced Concepts Team.

Contacts:

Victor Becerra
Tel: +44 (0118) 378 6703
Fax: +44 (0118) 378 8220
e-mail: v.m.becerra@reading.ac.uk

Dario Izzo
Tel: +31 (0)71565 3511
Fax: +31 (0)71565 8018
e-mail: act@esa.int

Ariadna id: 03/4101
Study length: 6 months.
Contract Number: 18138/04/NL/MV

Abstract

This report describes the analysis and development of novel tools for the global optimisation of relevant mission design problems. A taxonomy was created for mission design problems, and an empirical analysis of their optimisational complexity performed - it was demonstrated that the use of global optimisation was necessary on most classes and informed the selection of appropriate global algorithms. The selected algorithms were then applied to the different problem classes: Differential Evolution was found to be the most efficient.

Considering the specific problem of multiple gravity assist trajectory design, a search space pruning algorithm was developed that displays both polynomial time and space complexity. Empirically, this was shown to typically achieve search space reductions of greater than six orders of magnitude, thus reducing significantly the complexity of the subsequent optimisation. The algorithm was fully implemented in a software package that allows simple visualisation of high-dimensional search spaces, and effective optimisation over the reduced search bounds.

Contents

Chapter 1

Taxonomy of Mission Design Problems

This chapter presents a review of problems relevant to the area of mission design. It proposes a multi-dimensional taxonomy based upon problem characteristics and the complexity of the considered models. This taxonomy will be utilised in future during an analysis of the computational complexity.

1.1 Introduction

The gamut of mission design problems are of such variety that they cannot be appropriately classified into a simple 1-dimensional taxonomy. Consequently, this report attempts to classify different aspects of a mission design using problem constraints, control type, spacecraft model, solar system model and simulation accuracy.

The problem definitions given in the taxonomy consider mainly standard trajectory optimisation problems neglecting optimal attitude control, although extending it to more effectively encompass such attitude control problems would be possible.

1.1.1 Notation

The notation used in this report is as follows:

$\mathbf{X}$ Uppercase bold indicates a matrix.

$\mathbf{x}$ Lowercase bold indicates a vector.

$\dot{\mathbf{x}}$ A dot indicates the first differential of a variable with respect to time.

$f(t)$ A function of variable t.

1.2 Mission Design Problems

This section describes a list of mission design problems and their properties in ascending complexity. This list will then be used to create an efficient taxonomy for such problems.

1.2.1 Trajectory Design Problems

Point-to-point transfer

The most basic problem in trajectory design is creating a trajectory from one fixed point in an inertial frame (frequently heliocentric) to another. It is often assumed that the transfer is a Keplerian orbit, although non-Keplerian transfers are also possible.

Lambert's problem

If additional constraints are added such that the point-to-point transfer must occur in time t, and the transfer path is a Keplerian orbit, then this is known as a Lambert's problem [3, 10]. A single impulse is utilised to achieve the transfer orbit. This problem has a unique solution that must be obtained using numerical methods.

The parameters of a Lambert problem are the angle and magnitude of the thrust, which may create a two (2D astrodynamic model) or three (3D astrodynamic model) dimensional continuous search space.

Minimum Delta-V Lambert's problem

The Minimum Delta-V problem differs from a standard Lambert problem by removing the time constraint and instead minimising the amount of thrust required to achieve the manoeuvre [37, 10]. The problem dimensionality is the same as for the standard Lambert's problem as it is usually solved with a single impulse.

Orbit-to-orbit transfer

In orbit-to-orbit transfers it is usual to assume Keplerian orbits for both initial and final orbits.

Hohmann transfers

Hohmann transfers are bi-impulsive manoeuvres and were developed as an efficient way of transferring between circular orbits [20, 2]: the corresponding expressions for the appropriate change in velocity, $\triangle\mathbf{V}$, have a closed form and hence no search is necessary. Such transfers have since been extended to arbitrary elliptical orbits [36] resulting in a function of only one independent variable whose first derivative is also known, thus allowing simple optimisation [10].

Continuous thrust transfer

Another problem of interest in orbital transference is that of the minimum-time continuous thrust orbit transfer, where the transferral orbit is obviously non-Keplerian.

Body-to-body transfer

A body-to-body transfer is the most complex of the standard trajectory design problems. A currently popular problem of this type is the design of Earth-Mars trajectories [19, 47]. Because of the periodic motion of the planets multiple local minima exist and thus global search techniques (for example, genetic algorithms [21, 13] or estimation of density algorithms [32]) must be employed. The dimensionality of such problems relies on the astrodynamic model, the type of control applied and the technique used to optimise.

Gravity assists

Gravity assist manoeuvres (GAs) are frequently used to reduce $\triangle\mathbf{V}$ requirements of mission and also reduce mission duration [11]. A method suggested for using multiple GAs to create resonant orbits was suggested in [45] using a combinatorial approach to select putative swing-by sequences. Obviously the selection of GA sequences introduces some discrete dimensions into the optimisation problem with many local minima and hence global techniques not reliant on gradients must be used. The dimensionality is affected by both the number of possible GAs and the type of control employed by the spacecraft.

Ultimately, the number of swing-bys could be itself a parameter to optimise, hence resulting in a search space of varying dimensionality.

1.2.2 Weak Stability Boundary Problems

Lagrange points

Lagrange points are points in space where the gravitational attraction of two massive bodies rotating around their centre of mass combine to form a point where a third body of negligible mass would remain stationary with respect to the rotating frame. However, Lagrange points are (in dynamical system terms) repellers and, therefore, small course corrections are constantly required to remain in a Lagrange point

Weak Stability Boundary transfers

Weak Stability Boundary transfers exploit the invariant manifold structure associated with the Lagrange points of a pair of massive bodies, with research has mainly concentrating on modelling the primaries using the Planar Circular Restricted 3 Body Problem (PCR3BP) [27]. For two coupled three body systems, such as the Sun/Earth/Craft and Earth/Moon/Craft systems, there are regions where the invariant manifold structure of the Sun/Earth Lagrange points interacts with the invariant manifold structure of the Earth/Moon Lagrange points [27, 28]. Applying a small $\triangle\mathbf{V}$ at an appropriate point can therefore transfer from one manifold structure to the other, allowing a more fuel efficient transfer than would be possible by a standard Hohmann transfer (which are generally considered optimal).

One difficulty of modelling such trajectories that pass through or near Lagrange points is that a high degree of numerical accuracy is required when integrating the equations of motion. This is due to the sensitive chaotic dynamics that can occur in these regions [4].

1.2.3 Optimal Attitude Control

It has been shown that the problem of adjusting the orientation of a spinning gyrostat can be transcribed into a sparse non-linear programming problem and solved in the same way as trajectory design problems [24]. Therefore, the problem of optimal attitude control can be described generally as imposing boundary constraints in a similar way to a trajectory design problem, except they are on orientation and angular velocity rather than position and velocity.

Also, in trajectory design problems it is important to be able to perform effective reorientation manoeuvres on spacecraft in order to be able to apply thrust in a desired direction [24], and this requires the modelling of the spacecraft as a rigid body rather than as a point-mass. Ultimately, therefore, all space missions must consider the problem of optimal attitude control in order to manoeuvre, although good first guess solutions can be generated by using a point-mass spacecraft model. However, neglecting the rigid body dynamics can have a direct effect on the global optima of a problem as certain control strategies computed using the point-mass approximation may not be feasible in practise [51]. For example, large angular changes in thrust over a small time period may not be possible when using momentum wheels to adjust attitude.

Therefore considering attitude control adds additional constraints to the optimisation problem while increasing the number of state variables associated with the spacecraft (which if solved using collocation will significantly increase the problem dimensionality).

1.3 Mission Design Taxonomy

The primary function of this taxonomy is to facilitate the assessment of mission design problems in terms of their optimisational complexity, and therefore attention will be paid to how each aspect of a problem can affect this.

¿From the list of mission design problems in section 2 it is apparent that a simple taxonomy of such problems in terms of optimisation properties is not possible. Instead, it is more appropriate to classify different aspects of the problem and the models used, and examine how these could contribute to the difficulty of the optimisation. For example, the dimensionality of the problem is reliant upon the type of control (impulsive or continuous), the solar system model used (2D or 3D) and the method used to solve the optimal control (collocation or single shooting [5]).

The five aspects of each mission design problem are the problem constraints, control type, variables to optimise, spacecraft model and astrodynamic model.

A summary of the taxonomy presented in tabular form can be found in Appendix A.

1.3.1 Problem Constraints

The first stage is to identify the actual problem that must be solved independently of the quantities to be optimised i.e. what constitutes *any* valid solution to the given problem. These then define constraints on the boundary conditions such as the initial position $\mathbf{x}_i$, velocity $\dot{\mathbf{x}}_i$ and time t_i and the final position $\mathbf{x}_f$, velocity $\dot{\mathbf{x}}_f$ and time t_f. For instance, a transfer between two Keplerian orbits can be seen as imposing two constraints $f(\mathbf{x}_i, \dot{\mathbf{x}}_i) = 0$ and

$g(\mathbf{x}_f, \dot{\mathbf{x}}_f) = 0$: f and g are appropriate functions enforcing the constraint on the position and velocity required to maintain a Keplerian orbit. A Lambert's problem, by comparison, imposes $\mathbf{x}_i = \mathbf{a}, t_i = b$, $\mathbf{x}_f = \mathbf{c}, t_f = d$, where $\mathbf{a}$, b, $\mathbf{c}$ and d are constants.

Consequently, each mission design problem imposes its own constraints and it is the number of such constraints and their complexity which will determine the complexity of the optimisation required to satisfy them. These constraints can be divided into boundary constraints (initial and final) and continuous constraints.

For missions consisting of multiple phases the complexity of each phase can be classified separately (such as a series of GAs and deep-space manoeuvres), although achieving a global minima over the entire mission will be more complex than optimising each given phase alone.

Initial constraints

Initial constraints may be applied to t_i, $\mathbf{x}_i$ and $\dot{\mathbf{x}}_i$. In many cases one or more of these may be a function of another. The possible constraints are described below:

t_i may be constant (such as in a Lambert's problem) or unconstrained (not used in optimisation or is a variable to be optimised).

$\mathbf{x}_i$ may be constant, unconstrained or a function of time (starting in a given orbit).

$\dot{\mathbf{x}}_i$ may be constant or a function of time (starting in a given orbit).

Final constraints

Final constraints are applied to t_f, $\mathbf{x}_f$ and $\dot{\mathbf{x}}_f$. In many cases one or more of these may be a function of another. For example, if transferring to an orbiting body then the final position is a function of time $\mathbf{x}_f = f(t_f)$. Possible constraints on the final conditions are:

t_f may be constant (such as in a Lambert's problem) or unconstrained (not used in optimisation or is a variable to be optimised).

$\mathbf{x}_f$ may be constant, unconstrained, a function of time (orbiting body) or be a joint constraint with velocity (orbit).

$\dot{\mathbf{x}}_f$ may be constant, unconstrained or a joint constraint with velocity (orbit).

Other constraints

Constraints can also act continuously over a mission or at discrete points. For example, a continuous constraint may be to maintain appropriate satellite attitude or apply thrust to remain in a Weak Stability Boundary Orbit. The interdependence of such constraints can be expressed through the use of an appropriate function f:

$\mathbf{x} = f(t)$ Remain within a Lagrange point.

$\theta = f(\mathbf{x})$ Maintain attitude towards a fixed point in a given inertial frame.

$\theta = f(\mathbf{x}, t)$ Maintain attitude towards a given orbiting body.

Tabular constraint definition

The initial and final boundary constraints can be succinctly defined in a tabular form using the following notation:

Table 1.1: Constraints applied to some example problems in tabular form.

t_i	$\mathbf{x}_i$	$\dot{\mathbf{x}}_i$	t_f	$\mathbf{x}_f$	$\dot{\mathbf{x}}_f$	Problem description
#	#	#	#	#	#	Lambert's Problem
U	O	O	U	O	O	Orbital transfer
B	$f(t)$	$f(t)$	U	$f(t)$	U	Orbit-to-body transfer within a given launch window

# Constrained to a single value.

B Bounded within given limits.

U Unconstrained variable or is variable to be optimised.

O Orbit - This indicates a joint constraint between the position and velocity of the spacecraft at the corresponding boundary.

$f(t)$ Function of time.

Using this notation the boundary constraints of the mission design problem can be concisely defined. To illustrate this, Table 1.1 demonstrates the definition of some of the problems discussed in section 2. However, problems such as Weak Stability Boundary orbits are difficult to define explicitly as they rely heavily on the astrodynamic model in use. If only two body dynamics is used in an orbit transferral problem then the optimal solution will be an appropriate Hohmann transfer, whereas by using three body dynamics it becomes a ballistic capture using the Earth-Sun Lagrange point. Ideally, a global optimisation technique would therefore locate solutions using WSB without the need for explicit targeting (adding an additional constraint to ensure the trajectory passed through the Lagrange point).

1.3.2 Control Classes

The thrust that can be applied to a spacecraft can be divided into two main classes: impulsive and continuous thrusts.

Impulsive manoeuvres

An impulsive manoeuvre is one that is assumed to approximate an instantaneous change in momentum of the spacecraft, although of course in practise this change must take place over non-zero time.

Fixed *n*-impulse maneouvres

This control model utilises a fixed number of impulses where the parameters are the time, angle and magnitude of the impulses. The corresponding search space is therefore continuous and of dimensionality $3n$ (2D model) or $4n$ (3D model).

For example, a Hohmann transfer is a bi-impulsive manoeuvre used to complete an orbit transfer between two coplanar circular orbits, whereas Lambert's problem (transfer from point A to point B in time t) is solved using a single impulse to put the spacecraft into the appropriate orbit. Tri-impulsive manoeuvres have been shown to be more efficient than Hohmann transfers when transferring to much larger orbits.

Variable *n*-impulse manoeuvres

In this control model the number of impulses itself becomes a parameter, thus introducing a combinatorial element into the optimisation. Consequently, the dimensionality of the search space could vary during the optimisation and additionally produces a non-continuous search space, thus constraining the possible optimisation algorithms.

Continuous thrust

Usually a low power drive such as ion propulsion, the angle of the control over time is the control variable. As a consequence, continuous thrust intrinsically has infinite dimensionality, unlike impulsive control, and so methods must be employed to approximate this control curve. The simplest way to achieve this is to use n discrete time slices, where $\mathbf{u}_i$ is the control vector at time slice i. However, in order to obtain smooth control a large n is required and therefore the dimensionality is increased proportionally.

Alternatively, the control curve can be modelled as a polynomial such as a spline, in which case the dimensionality of the problem may be effectively reduced, at the expense of perhaps not being able to reproduce the globally optimum curve dependent on the polynomial order selected.

Continuous thrust can be further subdivided into *simple* and *complex* optimisation cases dependent on the propulsion system.

Simple continuous thrust

Simple continuous thrust is defined as a propulsion system that can exert the same thrust magnitude independently of position and attitude. Example of simple continuous thrust are RTGs (radioisotope thermal generator) or ion propulsion engines.

Complex continuous thrust

Complex continuous thrust is provided by propulsion systems such as solar sails [14, 41], where the available thrust magnitude and direction is constrained by position and attitude relative to the sun, thus increasing the complexity of the control optimisation.

1.3.3 Decision variables/objectives

This section describes the variables that are used as either decision variables or objectives during the optimisation of a mission design problem

Time

In manned missions, minimising aspects of mission time is essential due to radiation exposure and the amount of supplies required by the crew. In probe design mission time is less crucial although pre-bounded limits are usually set on the trajectory optimisation. Time can be both an objective and/or a decision variable: when optimising multiple gravity assist trajectories, the decision vector can be phrased as the time between successive planetary encounters, and it may also be desirable to minimise the mission time.

Thrust

In probe design, the mission time is generally less important than minimising the overall propellant required, and therefore the overall thrust $\triangle\mathbf{V}$ required by the mission. Thrust is normally used as an objective: less thrust equates to a larger scientific payload for a mission.

Velocity

An example of using velocity as an objective would be to minimise the swing-by velocity of a body of interest so that more measurements can be obtained. Although similar to optimising thrust, larger velocities do not affect the amount of propellant required and therefore the payload mass of the spacecraft.

Angular Momentum

When considering optimal attitude control it is useful to utilise total angular momentum as an objective. Minimising this quantity over a manoeuvre will also minimise the cost of propellant required by attitude thrusters or energy required by momentum wheels.

Multi-objective optimisation

Research is currently ongoing designing manned missions to Mars, where characteristics such as flight trajectories, energy requirements, travel times, and surface stay times are all variables that are desirable to optimise. This leads to the creation of *pareto fronts* in the search space representing families of optimal solutions with different weighting to the corresponding objectives [19]. Performing a global optimisation on a given weighting of multiple objective function corresponds to a given point on a pareto front, and hence it is beneficial to consider the front as a whole as more mission design options become apparent.

1.3.4 Spacecraft Model

During mission design, the spacecraft can be modelled at varying levels of complexity, corresponding to varying levels of optimisational complexity. The point-mass spacecraft may be the simplest model, but optimal solutions generated using it may be found to be non-optimal when considering controlling the attitude and angular momentum of the spacecraft.

Point-mass model

The point-mass model is the simplest way of modelling the spacecraft and is frequently used for the generation of first guess solutions. Any thrust applied is in the global frame of reference rather than being dependent on the state of the spacecraft.

Rigid body model

With the rigid body model the inertia of the spacecraft must also be taken into account, and consequently control must be calculated relative to the attitude of the spacecraft and its angular momentum. Obviously, in order to consider optimal attitude control problems the model must be at least of this complexity. The rigid body assumption in effect constrains the angle of thrust that can be applied in a given time period, and thus the feasible set of trajectories is a subset of those found using the point-mass model.

Low level control model

The highest level of modelling complexity considers optimising the torque required to control the spacecraft's angular momentum, thus allowing the design of minimum-torque manoeuvres.

1.3.5 Astrodynamic Model

This section describes the varying complexities of astrodynamic models that may be applied in mission design problems and their impact on the properties of the optimisation of such problems.

n-body dynamics

The simplest astrodynamic model is the restricted 2 body problem, where the mass of the spacecraft is assumed negligible and only the gravitational field of the orbited body (usually the Sun) is considered. However, in order to use Lagrangian points or design a Weak Stability Boundary orbit at least 3 body dynamics must be used. The Sun/Earth/Moon/Spacecraft 4 body system can be effectively approximated as two coupled 3 body systems [28]. Increasing the number of considered bodies can affect the optimisation by the generation of additional minima - with 2 body dynamics the Hohmann transfer is the most efficient way achieve a Moon orbit, but by including the Moon and Sun additional minima are created through the use of the Lagrange points. Also, using 3+ body dynamics allows the use of Gravity Assists that produce new global minima, although it is best to take these into account explicitly using the link-conic method [46] rather than integrating the state equations.

Planar/non-planar models

The orbital inclination of the planets relative to Earth's orbital plane differs by less than four degrees in most cases (Mercury and Pluto being the exceptions). Consequently, when considering planet-to-planet transfer a coplanar model is generally sufficient to generate good first guess solutions. 3D models increase the number of parameters to specify an angle from one to two and the amount of state variables from four to six (assuming point mass spacecraft), and thus increases the complexity of the optimisation.

Circular/elliptic orbits

Using circular orbits is a simple method of testing the efficacy of techniques, but in general is an inaccurate way of modelling the solar system. As an example, a popular design task is the creation of trajectories between Earth and Mars: using circular orbits each opposition will be at the same distance (approx. 78 million km), but by taking into account Mars' orbital eccentricity (0.0934) the opposition in August 2003 was only a distance of 56 million km. However, calculating elliptic orbits is more expensive than circular ones as Kepler's equation must be solved using an iterative method such as Newton's.

In general, the transferral from circular to elliptic orbits will not alter the nature of the optimisation significantly: it just improves the accuracy of the located solution.

Other factors

Other factors that may be modelled to increase accuracy are atmospheric drag, nutation and non-uniform gravitational fields provide more accurate results but will not fundamentally alter the properties of the search space, merely perturb the positions of optima.

1.4 Discussion

This section provides a short discussion on the mission taxonomy.

1.4.1 Numerical Accuracy

For most preliminary mission optimisation problems the numerical accuracy has only to be high enough to preserve the basic qualitative properties of the trajectory such that it can act as a good first guess solution to a highly accurate optimiser such as DITAN. When considering WSBs, however, a much higher level of accuracy is required due to the sensitive chaotic dynamics that occur in such regions.

1.4.2 Dimensionality of the Search Space

The dimensionality of a given mission design problem cannot always be easily defined, as the technique used to solve the problem can radically alter the dimensionality. For example, a popular direct technique for trajectory optimisation is collocation [18]: this radically increases the problem dimensionality while simultaneously making the problem amenable to local optimisation tools such as sparse SQP [5].

Additionally, by considering variable impulse manoeuvres and variable length swing-by sequences the resulting transdimensional search problems would be difficult to solve using standard methods and therefore are worthy of further consideration.

1.4.3 Other Properties of the Search Space

Although it is not possible to characterise exactly the properties the search space of each problem this early in the investigation, some broad observations are possible. The simpler problems, such as planar orbital transfer and point-to-point transfer in 2 body dynamics, correspond to smooth continuous search spaces with few local minima, although singularities can be possible if planets are modelled as point-masses. The more complex problems, such as WSB transfers and interplanetary transfers, can contain multiple local minima and have a highly oscillatory (non-smooth) nature. The use of collocation to solve problems requiring global optimisation may not be ideal, as global methods may not be able to take advantage of the underlying sparsity in the way sparse SQP does, and hence using a single shooting method may be preferable in such cases.

1.4.4 Simultaneous Optimisation of Multi-complexity Models

An interesting method of optimisation that has received little attention in the literature is the automatic optimisation of trajectories using a hierarchy of model complexities. Instead, it is usual to optimise fully using a simple model that use that solution as a first guess for optimisation with a more complex model. As an alternative to this it may be possible to consider different levels of modelling as part of a single overall global optimisation. This might be achieved by using a form of distributed optimisation, in which a population of agents optimise solutions for simple models and gradually upgrade the corresponding modelling complexity, thus ensuring an overall global optima.

1.5 Conclusion

This report has described a general multi-dimensional taxonomy for mission design problems that can include the vast majority of commonly encountered situations. By considering the effect of each problem aspect on the resulting optimisation problems will facilitate further analysis into the complexity of optimising problems within this taxonomy.

Chapter 2

Analysis of Problem Complexity

This report presents an empirical and theoretical analysis of the complexity of some relevant mission design problems, thus facilitating the selection of appropriate optimisation algorithms to solve them. The type of each problem is classified using the taxonomy developed in Work Package 1, and its complexity classified according to the work of Törn.

2.1 Introduction

This report analyses the complexity of a representative selection of mission design problems, thus facilitating the selection of appropriate global techniques to solve them. The methodology described by Törn [44] will be the main tool to analyse problem complexity.

An analysis will be performed of three problems identified as being important - optimal bi-impulsive interplanetary transfer, optimal low thrust interplanetary transfer, and multiple gravity assist trajectories.

2.1.1 Notation

The following notation will be used throughout this report :

f^* a minimum value of objective function f;

$\hat{f}^*$ an estimate of f^*;

$\mathbf{x}$ a vector in $\mathbb{R}^n$;

$\dot{\mathbf{x}}$ the derivative of $\mathbf{x}$ with respect to time.

2.2 Analysing Complexity

The purpose of analysing computational complexity is to characterise the *intrinsic* difficulty of a given optimisation problem. Using this information an appropriate optimisation algorithm may then be selected.

2.2.1 Convexity

The idea of convexity is key in optimisation, in that for a convex function any local minima must also be a global minima, and therefore optimisation will be much simpler than for non-convex functions - a local optimisation algorithm will be sufficient.

Consider a function $f : \mathcal{A} \to \mathbb{R}$ where $\mathcal{A}$ is a non empty set in $\mathbb{R}^n$. f is convex over $\mathcal{A}$ if

$$f(\lambda\mathbf{a} + (1-\lambda)\mathbf{b}) \leq \lambda f(\mathbf{a}) + (1-\lambda)f(\mathbf{b}), \qquad (2.2.1)$$

for $\mathbf{a}, \mathbf{b} \in \mathcal{A}$ and for all $\lambda \in (0, 1)$. Geometrically, this can be interpreted as a line drawn between any two valid input vectors, $\mathbf{a}$ and $\mathbf{b}$, must not intersect with the function manifold over $\mathcal{A}$, as illustrated in Figure 2.1.

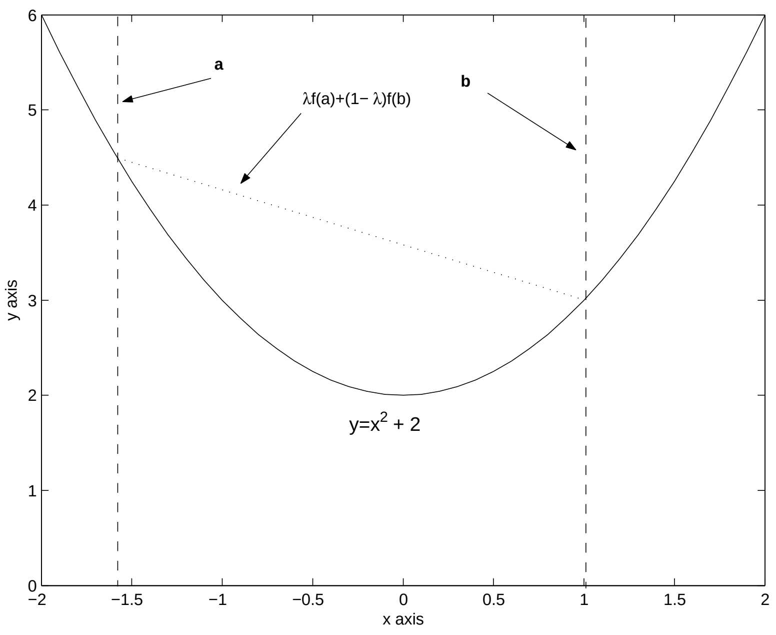

Figure 2.1: An example of a convex function illustrating the principle of the convexity criteria

Of course, if a function is non-convex over a set $\mathcal{A}$, it is likely to be convex over some subset of $\mathcal{A}$, unless the function has fractal properties. Similarly, as the size of the domain $\mathcal{A}$ is increased the convexity of a function can only *decrease*. Therefore, when considering non-linear functions it is important to optimise over a minimal domain in order to maximise convexity.

Unless the function has constant Hessian, or has a very special structure, convexity is not easily recognisable. Even for multivariate polynomials there is no known computable procedure to decide convexity [34]. Since it is known that many local optima exist in mission design problems due to periodic planetary motion, then over a large enough launch window the majority of mission design problems will be non-convex.

Due to these factors, it is judged that an empirical investigation of the difficulty of optimising relevant mission design problems would be more effective than theoretical analysis of the problems. Through the use of search space characteristics found in practical test cases, some qualitative estimate of the optimisational difficulty of the problems may be found. This classification can then be used to help select appropriate optimisation methods.

2.2.2 Methodology

This section describes the methodology recommended by Törn [44] for analysing the complexity of unconstrained global optimisation problems through the use of empirical investigation. The goal is to find

$$\hat{f}^* = f(\hat{\mathbf{z}}^*), \tag{2.2.2}$$

where

$$\hat{\mathbf{z}}^* \in \mathcal{A} \subset \mathbb{R}^n \tag{2.2.3}$$

so that the obtained minimum is within some tolerance of the global minimum f^*

$$|\hat{f}^* - f^*| < \epsilon. \tag{2.2.4}$$

As theoretical analysis of general problem complexity is, in general, prohibitively difficult, Törn recommends empirical investigation of problems in order to determine their difficulty. For example, the size of the domain $\mathcal{A}$ can have a major effect. The four features identified as being important for problem complexity are:

- Size of the basin of attraction of the global optimum.
- The affordable number of function evaluations.

- Embedded or isolated global minima.
- Number of local minimisers.

Size of the basin of attraction of global optimum

The size of the basin of attraction of the global optimum can be found by sampling uniformly within the domain $\mathcal{A}$ and using a strictly descending local minimisation with an infinitely small step to find the corresponding minimum. This will then give an approximation of the proportion of the domain that the basin of attraction fills. Although such a minimisation is impossible in practice, all local optimisation algorithms will give some approximation of this underlying objective function geometry. The local optimiser that will be used in this report will be Sequential Quadratic Programming (SQP) [5, 18], as it has been previously utilised very successfully in the solution of sparse mission design problems.

Note that the relative size of the basin of attraction of the global minimum is inversely proportional to the size of the domain $\mathcal{A}$, and therefore a minimal search domain should always be selected to maximise this proportional size, as well as to maximise convexity. For instance, if the domain $\mathcal{A}$ is the box $[0,1]^n$, and the proportional size of the basin of attraction (relative to $\mathcal{A}$) is $\hat{p}^*$, then the probability of selecting a point within the basin is $1-\hat{p}^*$. If $\mathcal{A}$ is increased in size to $[0,2]^n$, then size of A is increased by a factor of 2^n and consequently the probability of selecting a point in the basin is reduced to $1-\frac{\hat{p}^*}{2^n}$. Therefore it is of significant importance to minimise the size of the search domain whenever possible.

Affordable number of function evaluations

Any global optimisation may be achieved in infinite time using a trivial uniform random search or an exhaustive method, but this is certainly not the best way of solving general optimisation problems. Therefore, there must be some number of function evaluations N_f that causes the optimisation to be considered intractable, although this is obviously subjectively dependent on the perceived acceptable timescale and computing resources that may be available, and therefore N_f cannot be defined except under stringent constraints.

However, if the overheads associated with the optimisation algorithm can be considered negligible compared to the expense of evaluating the objective function then it is valid to compare the number of function evaluations required by different optimisation algorithms.

Embedded or isolated global minima

Törn defines the embeddedness of a global minimum as the presence of local minimisers close to the global minimiser in the search space, such that locating one of these local minimisers may aid in locating the global minimiser. This is justified since any stochastic global optimisation method has a possibility of progressing through multiple local minima towards the global minimum. A global minimum with other local minima nearby is *embedded*, otherwise it is *isolated*.

The embeddedness of the global minimum between two different problems can be compared as follows. Firstly, normalise the search space to the unit hypercube, and then plot a histogram of the density of local minima as Euclidean distance away from the global minimum increases. The Euclidean distance of local minima is further normalised by a factor of $\sqrt{n}$, the hyper-diagonal of the normalised search space, such that search spaces of varying dimensionality can be directly compared.

The number of local minimisers

Although a global minimum surrounded by local minima can be beneficial, they nevertheless reduce the basin of attraction of the global minimum. Consequently, the more local minima a problem has the more difficult it is to globally optimise it.

As well as point minimisers, it is possible that valleys will be encountered, which represent an infinite number of local minimisers. Empirically, these can be identified as manifolds of local minima found using a gradient descent method, although it is non-trivial to identify whether the valley floor itself is smooth or oscillatory and therefore whether such a manifold represents a true valley or many adjacent point minima.

Classification of problem complexity

Table 2.1, replicated from [44], shows the system devised by Törn to classify optimisation problems as either easy, moderate or difficult based upon the number of minima, probability of sampling the basin of attraction of the global minimum and embeddedness of the global minimum. The system also includes some qualitative recommendations of the types of suitable optimisation algorithms - this will prove beneficial in Work Package 3, as it considers the selection of appropriate global optimisation algorithms to solve these problems. In the techniques column, + indicates that the technique is usable whereas ++ recommends that the main effort should be made using this technique. The three types of technique considered are local, global and adaptive. An adaptive technique is one that uses no gradient information but gradually samples more points in the regions where good solutions have been found, such as genetic algorithms [21]. Similarly, in the 'local' column *ld* shows a strictly local method would be superior whereas *gd* favours *global descent*: local improvement through adaptive sampling.

Unimodal is considered the easiest class to optimise, and may comprise both convex and non-convex problems. The difference between the 1 and 2 subscript for each difficulty class is the number of local minima the problem has: for problems with few local minima local descent should be used whereas global descent is required in the case of many local minima.

Table 2.1: Table summarising Törn's method for classifying the difficulty of optimisation problems

		Problem Features			Solution techniques		
Class	Complexity	$(1-\hat{p}^*)^{N_f}$	Embeddedness	#mins	glob	local	adapt
U	unimodal	0	none	1	(+)	++ld	+
E_1	easy	small	any	few	+	+ld	+
E_2		small	any	many	+	+gd	+
M_1	moderate	large	embedded	few	+	++ld	+
M_2		large	embedded	many	+	++gd	+
D_1	difficult	large	isolated	few	++	+ld	-
D_2		large	isolated	many	++	+gd	-

2.3 Earth-Mars Transfer

This section analyses the difficulty of optimisation of a thrust optimal Earth-Mars transfer including a braking manoeuvre at Mars. The control is bi-impulsive. The decision variables are the launch date, t_0 (in MJD2000) and t, the mission time, in days. This is the same problem as described in [47], although with different bounding constraints. The position of the spacecraft is denoted $\mathbf{x}$, and the velocity of the spacecraft $\dot{\mathbf{x}}$.

2.3.1 Classification of problem type

The problem can be classified in the taxonomy developed in Work Package 1 as follows:

Variable	**U**	**#**	**B**	$f(t)$	**O**
$\mathbf{x}_i$				X	N/A
$\dot{\mathbf{x}}_i$				X	N/A
t_i			X	N/A	N/A
$\mathbf{x}_f$				X	
$\dot{\mathbf{x}}_f$				X	
t_f			X	N/A	N/A
Additional Constraints					

Control Class	**Subclass**	**Check/Value**
Impulsive	Fixed n-Impulsive	2
Continuous	Simple	
Impulsive	Variable Impulsive	
Continuous	Complex	

Objective	**Check**
Thrust	X
Mission time	
Velocity	
Angular momentum	
Other variables	

Spacecraft Model	**Check**
Point-mass	X
Rigid body	
Low level control	

Astrodynamic Model	**Check/Value**
n-body dynamics (2+)	2
Non-planar model	X
Elliptic orbits	X
Other factors	

2.3.2 Thrust optimal transfer

The initial objective function f to be investigated is the total $\triangle\mathbf{V}$ required in order to effect the interplanetary transfer:

$$f = |\dot{\mathbf{x}}_i| + |\dot{\mathbf{x}}_f|, \tag{2.3.1}$$

where $\dot{\mathbf{x}}_i$ is the initial hyperbolic excess velocity (relative to Earth) and $\dot{\mathbf{x}}_f$ is the velocity relative to Mars on entry to Mars' sphere of influence. $\dot{\mathbf{x}}_i$ and $\dot{\mathbf{x}}_f$ are determined for a given t_0 and t by solving the associated Lambert problem [2, 17, 10] [1]. The bounds on the decision variables were chosen such that several synodic periods would be present. Multiple revolution solutions to the Lambert problem were not considered. The bounds were

- $t_0 \in [800, 3800]$ MJD2000
- $t \in [100, 400]$ days.

Figure 2.2 shows the underlying geometry of the objective function in the 2D search space. It can be seen that several local minima are present, and the objective function is quasi-periodic in nature due to the synodic period. If the orbits were circular and coplanar then all minima would be identical global minima, as in each synodic period exactly the same relative planetary positions would be evident. However, with the presence of eccentric orbits and non-coplanar orbits a single global optimum is evident over any practical launch window.

In order to examine the basins of attraction of the global optimum the search space was discretised into 50 sections in both dimensions. For each grid point, an SQP search was performed and the corresponding local minimiser found. Minima located within a Euclidean distance of 1.0 day were classed as identical, thus allowing the basins of attraction of each minimum to be identified. Using this technique, eleven unique minima were found. Figure 2.3 shows the basin of attraction associated with each minimum by identically colouring initial positions that correspond to the same basin of attraction. Minima with less than a relative basin of attraction size of less than 0.01 have been omitted as these tend to coincide with SQP failing to converge correctly.

[1] the solver utilised is currently unpublished work by Dario Izzo

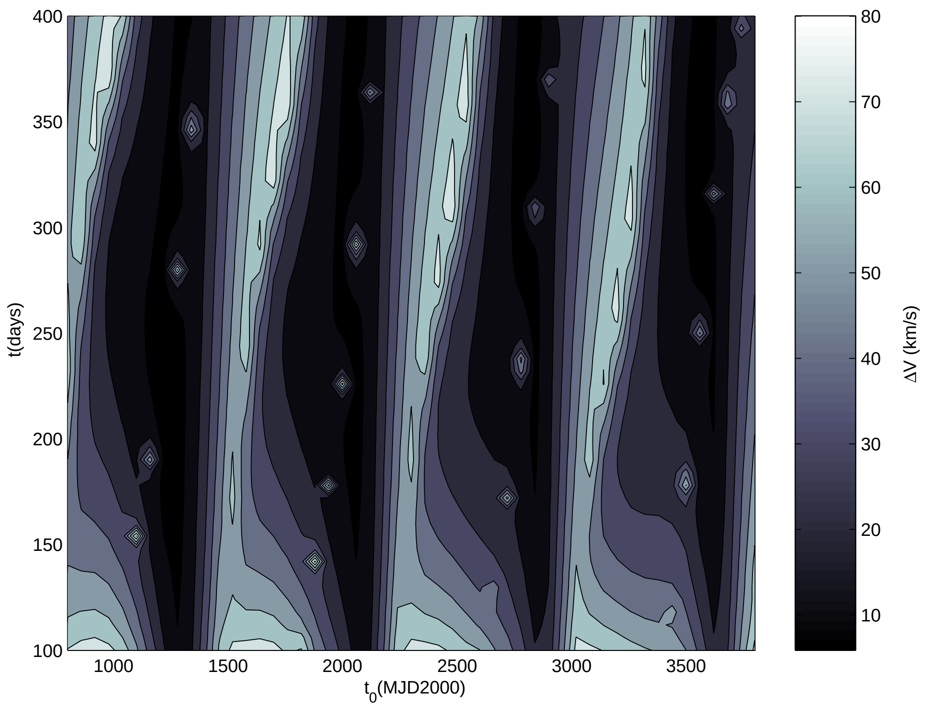

Figure 2.2: Objective function landscape of the Earth-Mars optimal thrust problem

Results

Table 2.2 shows the values of decision variables and objective function for all minima located with SQP. Note that minima 9 to 11 are not true minima, merely the result of the bounding constraints on the decision variables: if these bounds were relaxed these minima would no longer be evident. Figure 2.4 shows the globally thrust optimum trajectory.

Size of the basin of attraction of the global minimum

Table 2.3 shows the proportionate size of each minimum located within the search space. It is apparent that over this domain the global minimum (minimum 1) has a basin of attraction $\hat{p}^*$ corresponding to approximately 25% of the domain. This is associated with a small value for $(1-\hat{p}^*)^{N_f}$, the probability of not selecting a point within the basin of attraction of the global optimum after N_f uniformly random samples.

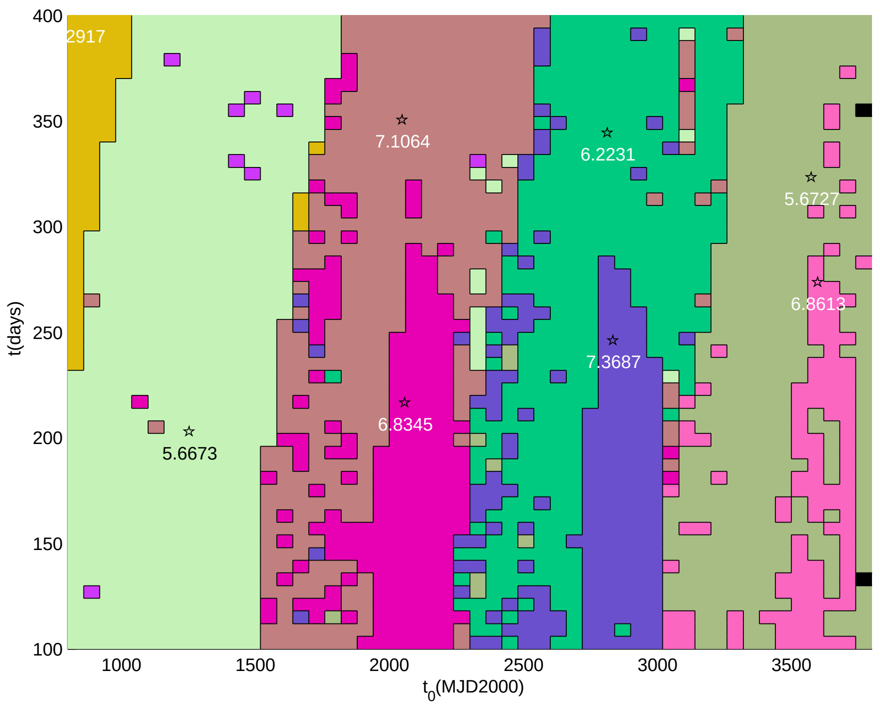

Figure 2.3: Basins of attraction for thrust optimal Earth-Mars transfer

Table 2.2: The decision variable and objective values of located minima in descending order of quality

Minima	$t_0(MJD2000)$	t(days)	$\|v_i\|$(km/s)	$\|v_f\|$(km/s)	f
1	1253.7	202.9	2.968	2.699	5.667
2	3573.4	323.2	3.207	2.466	5.673
3	2812.5	344.2	3.659	2.564	6.223
4	1225.5	233.2	3.539	2.816	6.353
5	2057.5	216.5	4.180	2.654	6.835
6	3597.8	273.5	4.364	2.497	6.861
7	2048.2	350.5	4.170	2.936	7.106
8	2834.3	246.0	4.768	2.601	7.369
9	3766.8	400.0	10.027	11.606	21.633
10	800.0	400.0	26.195	18.094	44.292
11	3800.0	126.7	37.877	25.439	63.289

Number of function evaluations

Using the information in table 2.2 it is possible to estimate the number of SQP searches, and therefore the approximate number of function evaluations required, in order to attain a 99% probability of having located the global minimum. The probability p_{fail} of not having selected a point in the basin of attraction of the global minima after k function evaluations is

$$p_{\text{fail}} = (1 - \hat{p}^*)^k, \tag{2.3.2}$$

and therefore to achieve a specific value of p_{fail} the required number of samples is

$$k = \frac{\log p_{\text{fail}}}{\log(1 - \hat{p}^*)}. \tag{2.3.3}$$

Calculating for this problem, a minimum of 16 SQP searches with random initial conditions would be required in order to achieve a 99% probability that the global optimum had been required by one or more of them. This result can also be generalised well to any given launch window: since four Earth-Mars synodic periods were included in the launch date and $\hat{p}^* \approx 1/4$ then, assuming the same global minimum, the relative size of the basin of attraction will be approximately $\hat{p}^* \approx \frac{1}{i}$, where i is the number of synodic periods included in the launch window. Therefore, an approximation of the

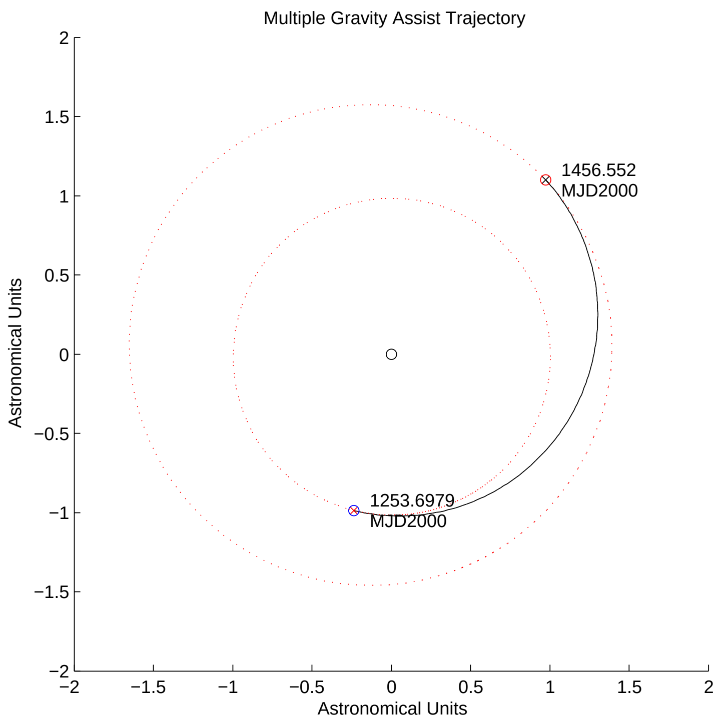

Figure 2.4: The globally optimum trajectory for a thrust optimal Earth-Mars transfer

Table 2.3: The approximate proportional size of the basins of attraction found using SQP

Minimum #	Basin proportion
1	0.2520
2	0.1680
3	0.1652
4	0.0032
5	0.0852
6	0.0476
7	0.1704
8	0.0820
9	0.0004
10	0.0256
11	0.0004

number of searches required is

$$k \approx \frac{\log p_{\text{fail}}}{\log\left(1 - \frac{1}{i}\right)}, \tag{2.3.4}$$

and since $\log(1-x) \approx -x$ when $0 < x \ll 1$, then

$$k \approx -i \log p_{\text{fail}}. \tag{2.3.5}$$

Hence the required number of SQP searches increases approximately *linearly* with the size of the launch window. Therefore, if the SQP parameters were initialised appropriately (in terms of step size) the computational expense of SQP may remain relatively constant with respect to the number of synodic periods considered, and hence the time complexity of this problem relative to launch window size s_w would be approximately $O(s_w)$.

Number of local minima

Over the selected launch window ten local minima have been identified. Following Törn's classification of the 'Shekel10' function [43], which also possesses 10 minima, it can be concluded that the Earth-Mars optimal thrust transfer belongs to a family of optimisation problems with *few* local minima.

Embeddedness of global optimum

Figure 2.5 shows a histogram of the normalised distance of each local minimum from the located global minimum. It can be seen that there are no local minima very close to the global minimum, and therefore this minimum is *isolated*.

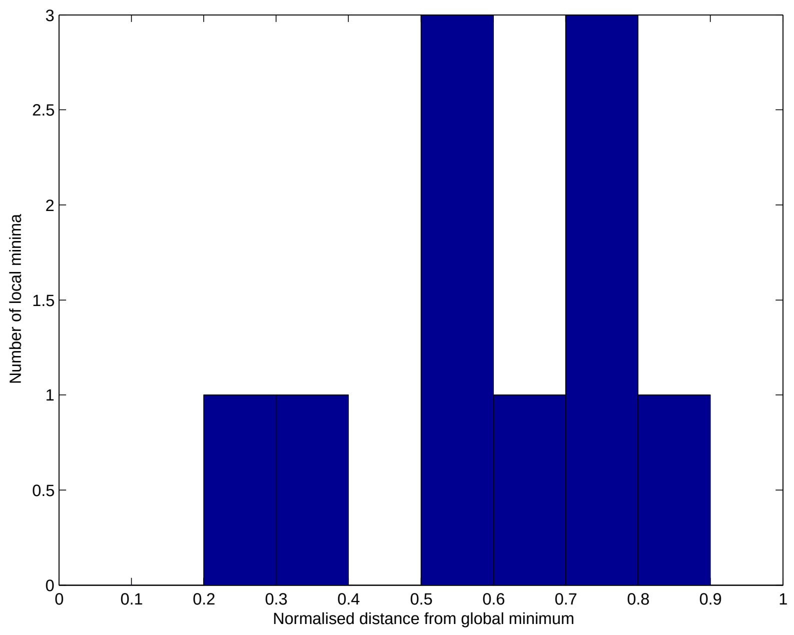

Figure 2.5: A histogram showing the number of local minimisers with respect to their normalised distance from the global minimiser

Classification of problem complexity

It has been established that $(1-\hat{p}^*)^{N_f}$ is small in this problem, the number of minima is few and the global minimum is isolated. This problem therefore belongs to class E_1 (see Table 2.1), and thus can be considered *easy* to optimise.

$(1-\hat{p}^*)^{N_f}$	Small
Embeddedness	Isolated
#mins	Low
Classification	E_1 **- easy**
Recommended technique	**Local**

2.3.3 Multi-objective optimisation

Obtaining minimal thrust on a mission is important, but is not the only factor to be taken into consideration in practice. Therefore, this section considers the analysis of a multi-objective thrust/mission time optimisation through the recovery of pareto fronts, thus allowing the recovery of a family of solutions rather than a single point. For a set of objectives where $f_i(\mathbf{z})$ returns value of the i^{th} objective function, the pareto front is the infinite set of values of $\mathbf{x}$ whose objective function values are not dominated by any other. Assuming minimisation, for two points within the search space, $\mathbf{z}_1, \mathbf{z}_2 \in \mathcal{A}$, $\mathbf{z}_1$ dominates $\mathbf{z}_2$ if $\forall_i f_i(\mathbf{z}_1) < f_i(\mathbf{z}_2)$.

An approximation of the pareto fronts can be found in the following way:

```
Grid sample the objective function to form a set $\mathcal{S}$
Let $\mathbf{x}$ be the first item of $\mathcal{S}$.
```

```
Repeat
    Disregard any members of $\mathcal{S}$ that are dominated by $\mathbf{x}$.
    Let $\mathbf{x}$ be the next item of $\mathcal{S}$.
Until (all members of $\mathcal{S}$ have been considered)
```

Another way of considering multi-objective optimisation in this case is as a weighted scalar objective function of the form

$$f = a(|\dot{\mathbf{x}}_i| + |\dot{\mathbf{x}}_f|) + bt, \tag{2.3.6}$$

where the ratio of a and b determine the relative importance of the objectives.

Figure 2.6 shows the approximation of the pareto fronts found using a grid sampling density of 100 in both dimensions. However, unlike most illustrations of pareto fronts, one of the objectives is also one of the decision variables in this case (mission time), and hence the pareto front tends towards the bottom of the graph. The pareto front describes a line at $t = 100$ days, corresponding to $a = 0$ (no weighting on thrust), due to this coupling between decision variable and objective.

An approximately vertical line can be observed from the $t = 100$ line to the thrust-optimal global optima as $b \to 0$. Therefore, it is clear that regardless of whichever weighting between the objectives is desired, the search space of interest lies in a small portion of the originally defined decision variable bounds. Figure 2.7 shows a 3D representation of the optimal thrust manifold, with black crosses marking those points found to be on the pareto front - as expected, they traverse down the deepest part of the valley from the optimal time solution ($t = 100$ days) to the optimal thrust solution.

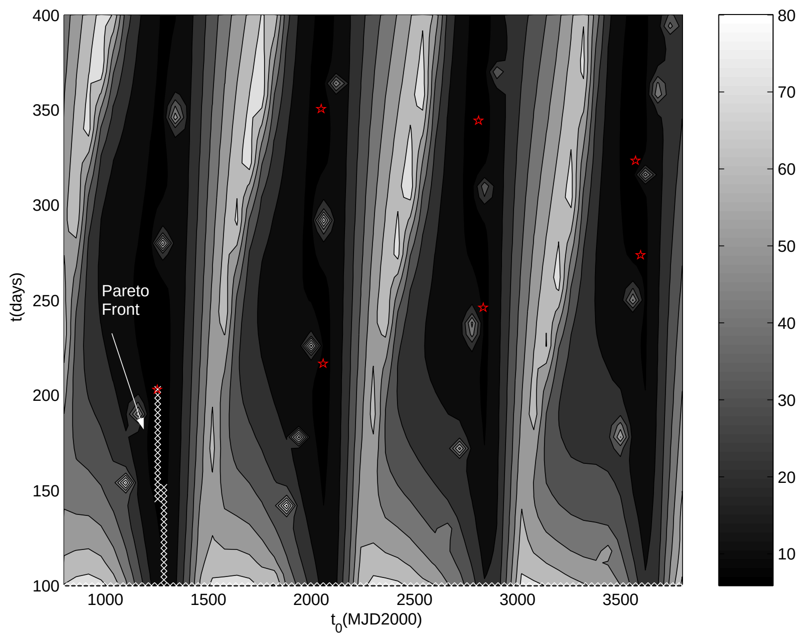

Figure 2.6: A graph of the thrust/mission time pareto front overlaid onto a contour plot of the thrust objective function. The optimal thrust minima are shown as red stars.

2.3.4 Discussion

This section has examined the complexity of finding an optimal thrust bi-impulsive interplanetary transfer from Earth to Mars. By examining the number of minima and the size of the basin of attraction of the global optimum, it has been determined that the difficulty of optimising such a problem is low. A multi-objective optimisation was then considered between thrust and mission time and the Pareto front was identified as being small and simple to locate.

2.4 Gravity Assist Trajectories

Gravity assist manoeuvres provide an effective way of reducing the propellant required for a mission, although usually at the cost of increased mission times [30, 45, 48]. This makes them ideal for non-manned missions to the

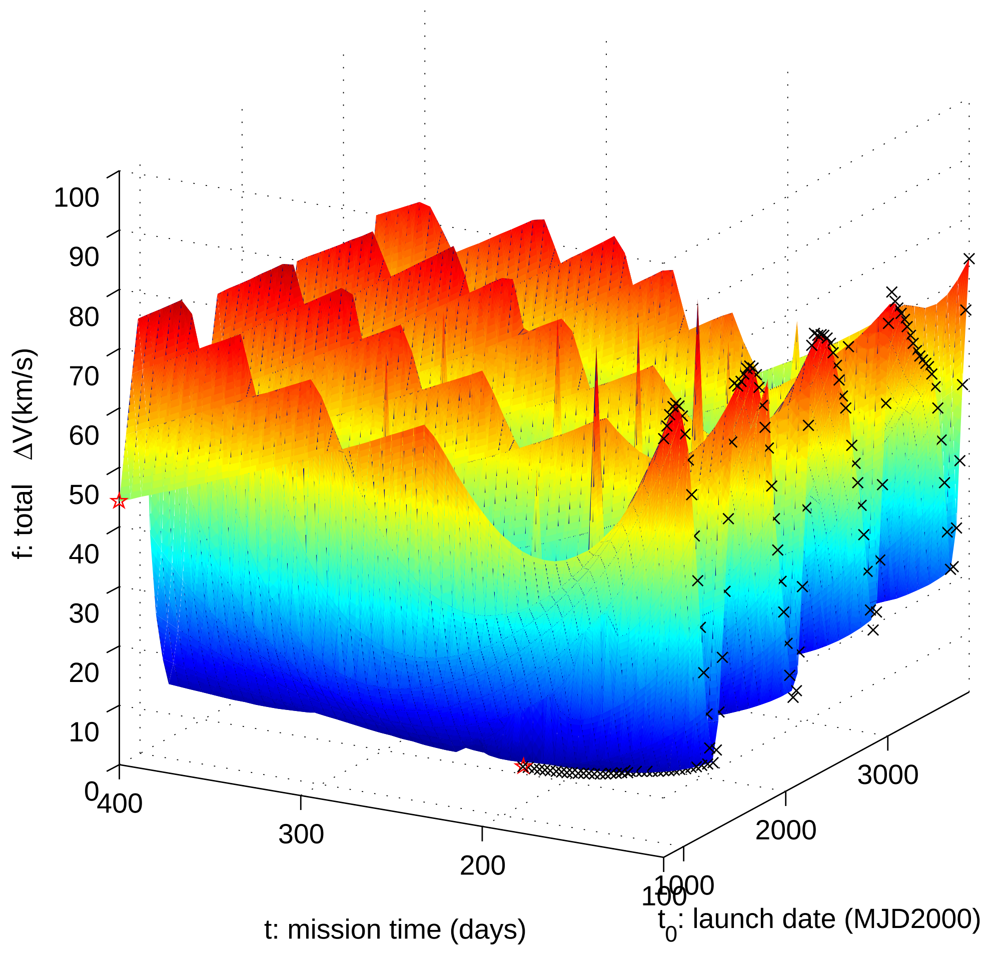

Figure 2.7: A graph of the thrust/mission time pareto front overlaid onto a 3D surface of the thrust objective function. The optimal thrust minima are shown in red.

outer planets that would be infeasible using a standard Hohmann manoeuvre [20, 37]. Also, the optimisation of swing-by sequences is of particular interest as it combines both combinatorial and continuous optimisation.

For this report, a simplified version of the Cassini-Huygens Earth-Saturn mission will be considered as a test case, given that only unpowered swingbys will be allowed. Consequently, since the actual Cassini-Huygens mission required at least one significant gravity assist $\triangle\mathbf{V}$ during in the EVVEJS transfer, only the simplified EJS may be considered:

- 2 body dynamics (only gravitational attraction of the Sun is considered).
- Elliptic, non-coplanar orbits using analytical Ephemeris.
- Point-mass spacecraft assumption.
- Link-conic approximation for gravity assist (unpowered swingby)

The orbital injection will remain the same as the original Cassini-Huygens mission, with the following parameters:

- Eccentricity, $e = 0.98$
- Radius of periapse, $r_p = 1.0895 \times 10^5$km

Orbital injection is achieved with a single impulsive thrust at the periapse of the hyperbolic orbit, thus creating a highly eccentric elliptic orbit. The desired angular momentum h of the orbit can be calculated simply as follows by finding the semi-major axis a, semi-latus rectum l

$$a = \frac{r_p}{1-e} \tag{2.4.1}$$

$$l = \frac{a}{1-e^2} \tag{2.4.2}$$

$$h = \sqrt{lGM}, \tag{2.4.3}$$

where G is the gravitational constant and M is the mass of Saturn. The magnitude of the velocity at periapse, $\mathbf{v}_{pf}$, is then simply

$$|\mathbf{v}_{pf}| = \frac{h}{r_p}. \tag{2.4.4}$$

The magnitude of the velocity the probe will achieve at the periapse of the orbit, $|\mathbf{v}_{pi}|$ can be calculated simply from the transferral of potential energy to kinetic energy, so

$$|\mathbf{v}_{pi}| = \sqrt{|\mathbf{v}_o|^2 + \frac{2GM}{r_p}}, \tag{2.4.5}$$

and therefore the braking manoeuvre is of the magnitude

$$|\triangle\mathbf{V}| = |\mathbf{v}_{pi}| - |\mathbf{v}_{pf}| \tag{2.4.6}$$

2.4.1 Classification of Problem Type

Multiple Gravity Assist trajectories do not fit directly into the taxonomy developed in Work Package 1, although each interplanetary phase may be characterised in the same way as the bi-impulsive Earth-Mars transfer as discussed in Section 2.3. However, since each Gravity Assist adds another dimension to the search space of the problem, then each additional phase may be assumed to have a multiplicative effect on the problem difficulty.

2.4.2 Earth-Jupiter-Saturn

The mission to be used as the benchmark for Gravity Assist missions is the Cassini-Huygens mission. A launcher is to be used to provide the initial C3, although the probe itself must provide the thrust required to achieve orbital insertion at Saturn.

The mission was split into two sections, the Earth-Jupiter phase and the Jupiter-Saturn phase.

The decision variables were the launch date t_0, the length of the Earth-Jupiter leg, t_1 and the length of the Jupiter-Saturn leg, t_2. For each date the corresponding Lambert problem was solved. The decision vector $\mathbf{x}$ was therefore

$$\mathbf{x} = \{t_0, t_1, t_2\}. \tag{2.4.7}$$

The bounds on the decision variables were based on the transfer time of a Hohmann transfer assuming coplanar and circular orbits: the lower bound was 10% of this value and the upper bound was 300% of this value. The launch window was selected to be a period of several years including the actual Cassini-Huygens launch date of 15th October 1997. The overall bounding constraints were

- $t_0 \in [-1278, 546]$ days
- $t_1 \in [100, 2992]$ days
- $t_2 \in [366, 10981]$ days

An equality constraint was used to ensure that the magnitude of the incoming and outgoing velocity at Jupiter (relative to Jupiter) was the same, thus effecting an unpowered gravity assist. The minimum altitude allowed for a gravity assist was 10% of the planetary radius.

The objective function was grid sampled with a density of 50 in each dimension (125000 function evaluations overall). Figure 2.8 shows the feasible solution set in the 3D search space in terms of the solutions with a difference in velocity magnitude at the gravity assist of less than 100m/s. The large tolerance of 100m/s was used to compensate for the nature of the grid sampling which is unlikely to directly locate minima that fully satisfied the constraint. Additionally, each valid point is coloured according to the required hyperbolic excess velocity from Earth. It can be seen that five main, approximately planar, manifolds exist within the search space with increasingly optimal solutions as t_2 increases up to about 6000 days. These manifolds are, as expected, quasi-periodic as the launch date is varied due to the Earth-Jupiter synodic period, and the motion of Saturn causes the variation in each manifold. Other manifolds also exist for very high values of t_2.

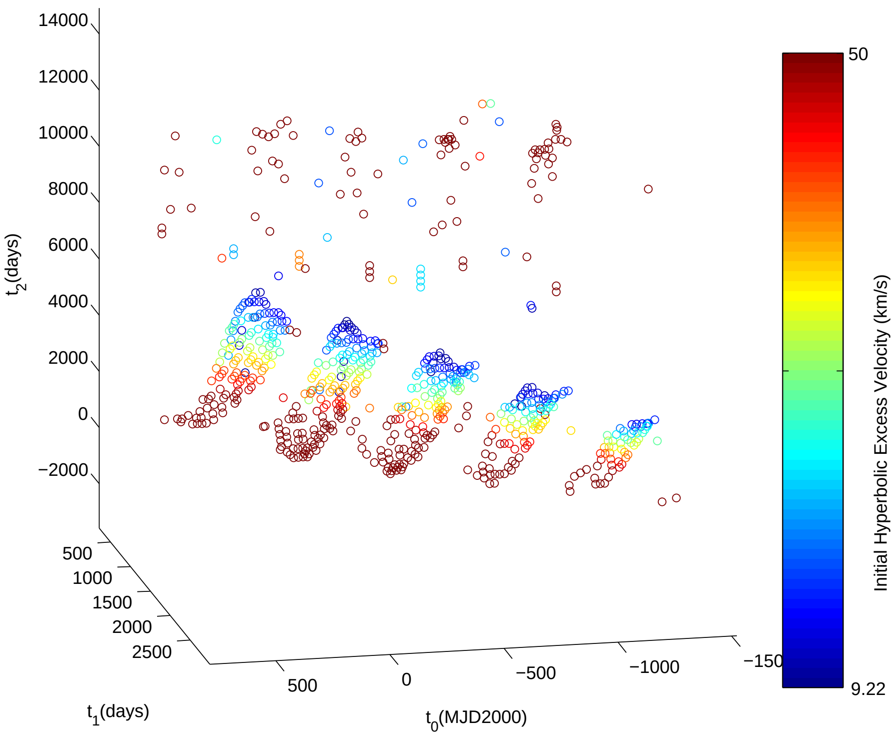

Figure 2.8: Manifold of valid solutions for an EJS unpowered swingby

2.4.3 C3 optimal trajectory

The first objective to be considered is the launch energy from Earth (the square of the required hyperbolic excess velocity).

The search space was sampled uniformly randomly and at each point a SQP search was performed, thus allowing the calculation of approximate basins of attraction. The number of trials performed was 5000.

Before the identification of minima, results which contained gravity assist errors were discarded. An error was defined as a swingby altitude of less than 110% of the planetary radius, or a discrepancy between incoming and outgoing velocity of greater than 10m/s.

Identical minima were then grouped using a tolerance of 10 for the Euclidean distance between the decision vectors. In total, 1302 unique optima were identified that satisfied the unpowered swingby criteria, although 1199 of these were found two or less times throughout the trials. From all the located optima, 296 had a C3 of less than 100km^2/s^2, which is feasible but limits the mass of the probe to approximately 300kg (based on the performance of an Ariane 44L launcher).

Table 2.4 shows the values of the decision variables and objectives for the forty best minima in terms of C3. Interestingly, it can be seen that the minima come in pairs, corresponding to an identical Earth-Jupiter transfer and then a short or a long Jupiter-Saturn transfer.

Figure 2.9 illustrates the positioning of the located minima within the search space. Again there is clear quasi-periodicity in the launch date t_0, and it is obvious that the search space consists of valleys rather than point minima, which could prove beneficial when using global optimisation rather than local optimisation if they can be traversed in an effective way.

Three mission options will now be presented, representing three of the most interesting minima - these were identified manually as minima 1, 83 and 272.

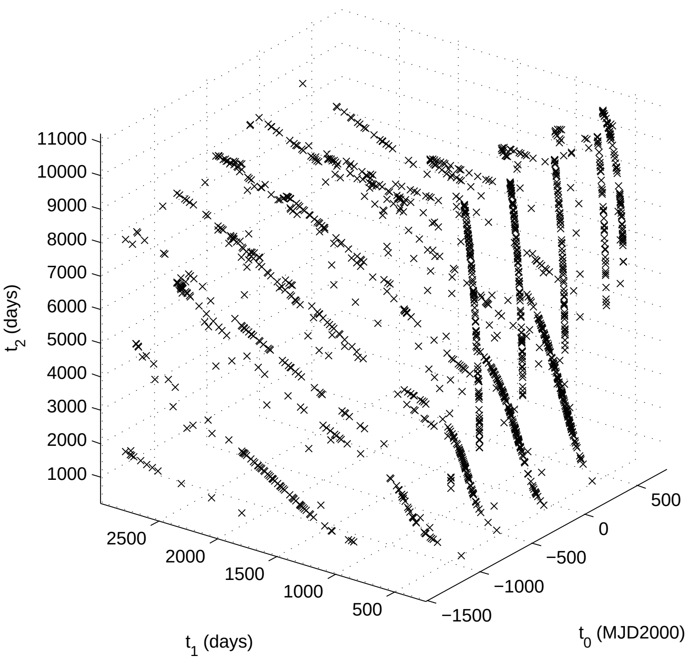

Figure 2.9: All the minima located using SQP in the EJS problem

Table 2.4: The forty best minima found by applying SQP with a randomly selected starting point over five thousand trials. The minima are presented in increasing value of C3

Minimum	t_0(MJD2000)	t_1(days)	t_2(days)	$\triangle\mathbf{V}_E$(km/s)	$\triangle\mathbf{V}_S$(m/s)
1	-177.2362	913.3063	4413.467	8.9179	432.1834
2	-177.3634	910.5667	4403.1539	8.918	432.0287
3	-177.3951	910.1847	6119.5224	8.9181	489.5624
4	-177.0969	917.0883	6133.418	8.9182	489.1924
5	-177.4859	907.65	4391.993	8.9185	431.8756
6	-177.7005	902.2847	6103.7275	8.9201	490.165
7	-177.8217	897.672	6094.6342	8.9219	490.6031
8	-177.9112	891.9186	4328.731	8.9248	431.2465
9	-175.9127	929.5323	4472.4369	8.9249	433.3292
10	-177.9778	890.2654	6080.1377	8.9257	491.4427
11	-178.3261	873.0997	4245.0416	8.9369	430.8767
12	-178.4967	866.6389	4214.1	8.9418	430.84
13	-178.599	861.6709	4189.6806	8.9457	430.8449
14	-178.7889	853.0536	4145.7216	8.953	430.9184
15	-178.845	849.8523	4128.9479	8.9558	430.9672
16	-178.9169	847.9958	4118.9198	8.9575	430.9996
17	-178.9554	845.5392	4105.7548	8.9597	431.0501
18	-173.1533	954.4039	6213.4049	8.9646	489.4496
19	-179.1572	837.3038	4060.0809	8.9676	431.267
20	-179.2849	833.2153	4036.5704	8.9717	431.4025
21	-179.2546	822.8718	3976.6917	8.9826	431.8341
22	-179.6337	817.6133	3943.1189	8.9885	432.0946
23	-171.4161	965.2712	4586.6657	8.9937	436.9473
24	-179.7507	813.0815	3914.5697	8.9937	432.3464
25	-179.9091	806.1342	3869.7324	9.0022	432.7764
26	-179.98	804.2098	5914.3338	9.0046	515.1292
27	-179.9023	799.1557	3824.1549	9.011	433.2602
28	-180.1451	793.2928	3783.3657	9.0189	433.7081
29	-180.2666	789.294	3755.179	9.0245	434.0343
30	-180.2965	786.1101	3732.7701	9.029	434.305
31	-180.5248	779.1051	5867.1603	9.0394	528.0789
32	-180.5413	776.3488	3661.2292	9.0436	435.2009
33	-180.5803	774.8725	3650.1375	9.046	435.3447
34	-180.5163	770.6256	3618.9376	9.0526	435.766
35	-180.6876	769.1977	3607.1309	9.0551	435.9158
36	-180.7227	766.1807	3583.9941	9.06	436.2312
37	-180.8309	761.0211	3543.4634	9.0689	436.791
38	-180.9428	755.1663	3496.5249	9.0794	437.4544
39	-180.999	753.1702	3480.1556	9.0831	437.6878
40	-181.1842	750.6304	3458.3838	9.0881	437.9933

Evaluation of minimum 1

The first minimum to be examined is the global minimum relative to the C3 objective, and has the following values of decision variables and objective values:

Decision Variables	
t_0(MJD2000)	-177.2362
t_1(days)	913.3063
t_2(days)	4413.467
Objectives	
Mission Time (days)	5326.7733
$\triangle\mathbf{V}_E$(km/s)	8.9179
$\triangle\mathbf{V}_S$(m/s)	432.1834
$\triangle\mathbf{V}_{GA}$(m/s)	0.00028098

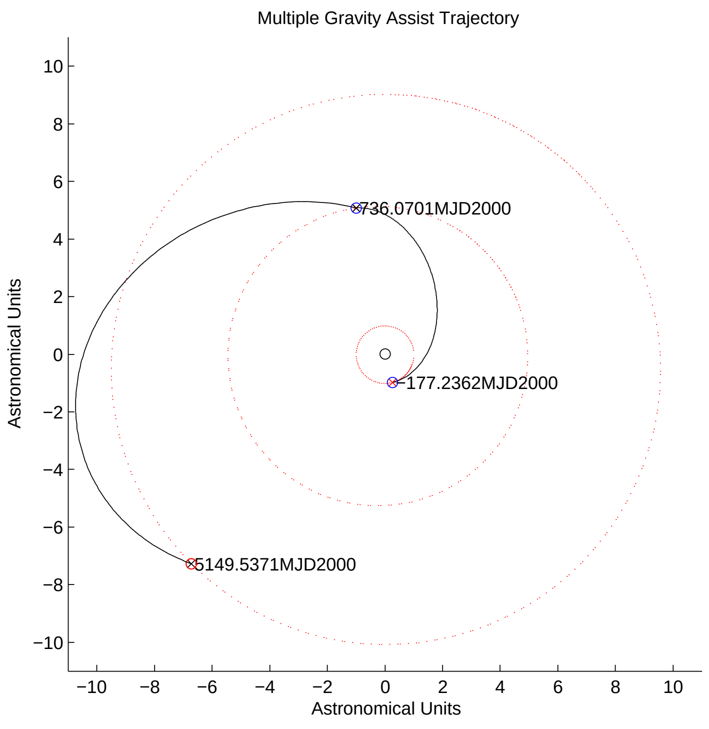

Figure 2.10: Global optimum for minimal launch C3

The trajectory associated with the global minimum is shown in Figure 2.10, and corresponds to a spiral orbit that travels significantly outside the orbit of Saturn before reaching apoapsis and proceeding to rendezvous.

Evaluation of minimum 83

Minimum 83 is worthy of individual consideration as it permits a much shorter mission time (only 4.3 years rather than 14.6 for minimum 1). However, although it requires only slightly more C3, significantly more thrust is necessary in order to achieve orbital insertion at Saturn (1.678km/s rather than 0.432km/s). Therefore, in this case, much more of the probe's weight will have to be fuel than for minima 1 and 2. It should be noticed that in some cases a large decrease in mission time might be considered an important improvement.

Decision Variables	
t_0(MJD2000)	-983.6982
t_1(days)	722.1952
t_2(days)	852.5393
Objectives	
Mission Time (days)	1574.7345
$\triangle \mathbf{V}_E$(km/s)	9.1881
$\triangle \mathbf{V}_S$(m/s)	1678.2072
$\triangle \mathbf{V}_{GA}$(m/s)	0.0011995

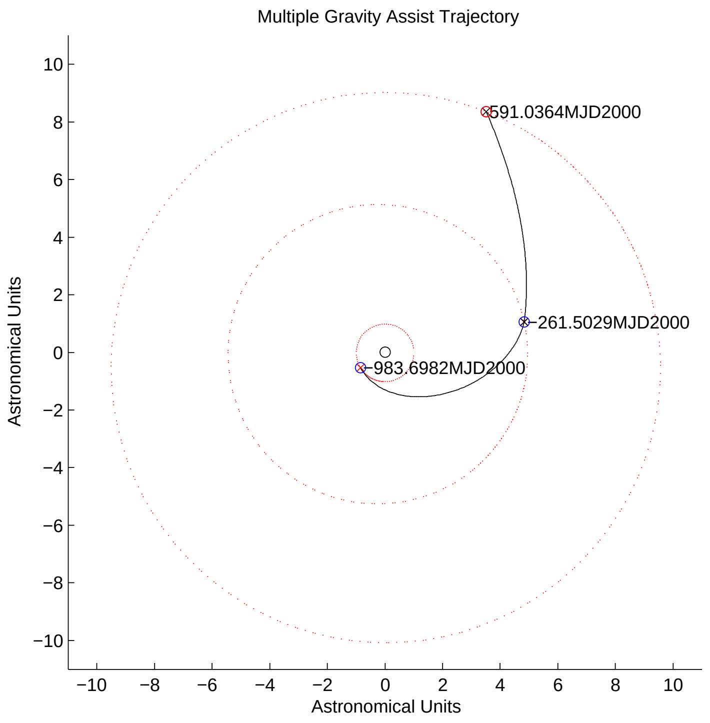

Figure 2.11: Trajectory plot for minimum 83

Evaluation of minimum 272

This gravity assist transfer is the orbital insertion thrust optimum, requiring a $\triangle\mathbf{V}$ of only 296m/s. However, the launch energy is higher, thus further decreasing the maximum mass of the probe, although it would depend of the performance of the launcher whether more payload mass would be available overall through this solution or minimum 1. Another advantage of this trajectory is that it is much shorter than minimum 1, at 2907 days rather than 5326 days.

Decision Variables	
t_0(MJD2000)	-965.435
t_1(days)	974.1727
t_2(days)	1932.4136
Objectives	
Mission Time (days)	2906.5863
$\triangle \mathbf{V}_E$(km/s)	9.7385
$\triangle \mathbf{V}_S$(m/s)	296.2322
$\triangle \mathbf{V}_{GA}$(m/s)	7.635e-005

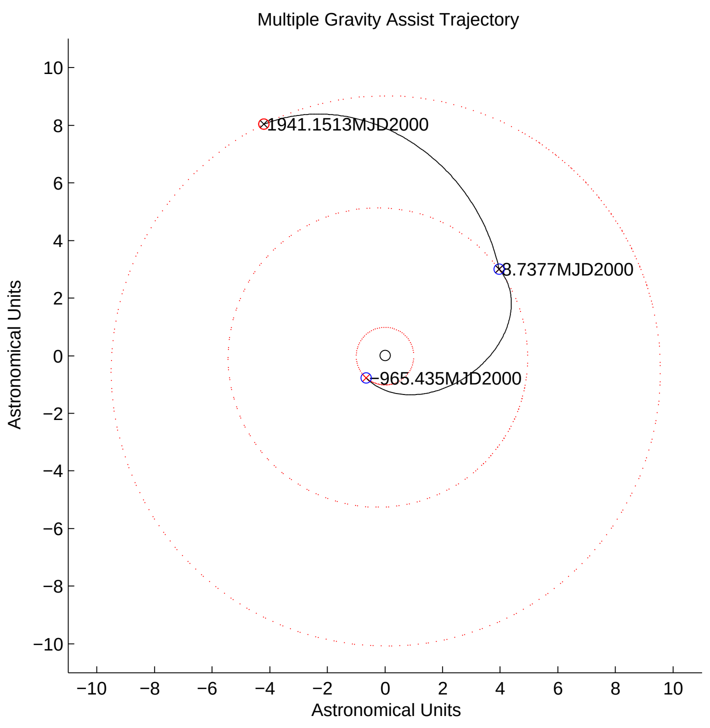

Figure 2.12: Trajectory plot for minimum 272

Basin of attraction of global optimum

Table 2.5 gives an approximation of the size of the basin of attraction $\hat{p}^*$ of each minimum. It can be seen that the majority of the near optimal minima have a very small basin of attraction, being located only one or twice over

Table 2.5: Table showing the proportional size of the basin of attraction for the 40 best minima located

Minimum	$\hat{p}^*$	Minimum	$\hat{p}^*$
1	0.077473	2	0.0014804
3	0.006415	4	0.0051813
5	0.00074019	6	0.00049346
7	0.00049346	8	0.00024673
9	0.00049346	10	0.00049346
11	0.00024673	12	0.00024673
13	0.00049346	14	0.00024673
15	0.00024673	16	0.00074019
17	0.00074019	18	0.00024673
19	0.00024673	20	0.00024673
21	0.00098692	22	0.00074019
23	0.00024673	24	0.00049346
25	0.00074019	26	0.00049346
27	0.00074019	28	0.00024673
29	0.0012337	30	0.00024673
31	0.00024673	32	0.00024673
33	0.00049346	34	0.00024673
35	0.00024673	36	0.00024673
37	0.00074019	38	0.00049346
39	0.00024673	40	0.00024673

the five thousand trials. However, the basin of attraction of the best solution found is comparatively large at 7.7% of the search space. Therefore, it can be said that $(1-\hat{p}^*)^{N_f}$ is *small.*

Number of function evaluations

Using a simple link conic model for the gravity assists negates the requirement of any numerical integration, and therefore the main source of computational expense for calculating the objective function comes from the two Lambert problems that must be solved. Hence, for n gravity assists the objective function is $n+1$ times as expensive as a single interplanetary transfer, and therefore the number of acceptable function evaluations for multiple gravity assist trajectories would be a fraction $\frac{1}{n+1}$ of the Earth-Mars case.

Number of local minima

The minimum number of local minima under these bounds is 1302. Following Törn classification of the Griewank2 function [43] (which has 500 local minima) as containing *many* local minima then the EJS problem must be classified similarly.

Embeddedness of global minimum

Figure 2.13 shows a histogram of the normalised distance of the local minima from the global minimum. It is noticeable that there are no minima greater than a normalised distance of 0.6 from the global optimum, but this is simply explained by the fact that the global optimum is located near the centre of the search space volume. A large number of optima are apparent close to the global optimum, undoubtedly due to the manifolds observed in Figure 2.9. Therefore, the EJS problem can be considered to have an *embedded* global optimum.

Classification of problem complexity

¿From this analysis, it is apparent that the EJS problem is more difficult to optimise than the Earth-Mars trajectory, the differing factor being the much larger number of optima within the search space.

$(1-\hat{p}^*)^{N_f}$	Small
Embeddedness	Embedded
#mins	High
Classification	**E_2 - easy**
Recommended technique	**Local with global descent**

From this classification, the recommended technique for solving this type of problem is not SQP, but a global optimisation technique that is able to

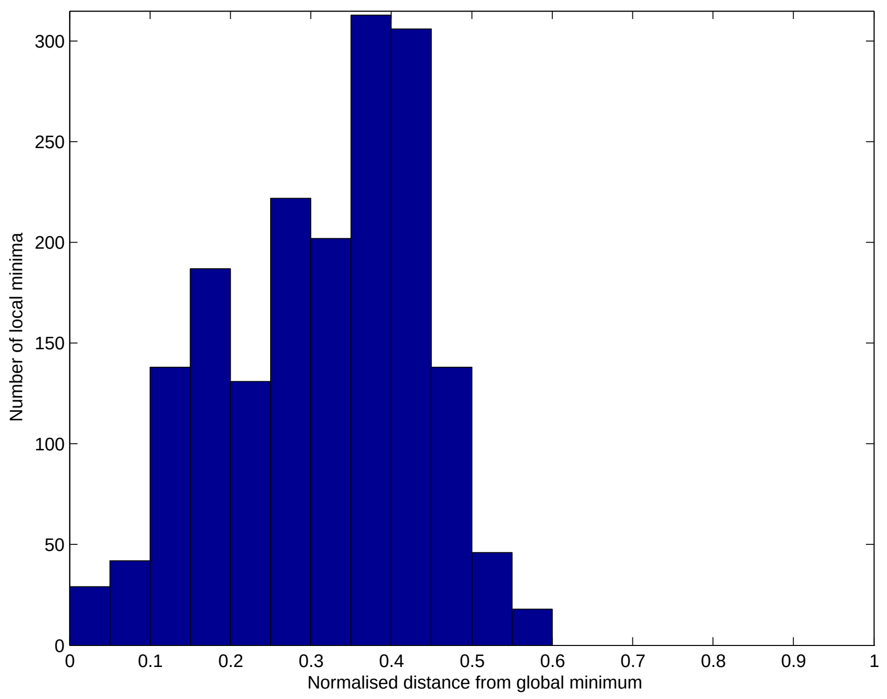

Figure 2.13: Histogram showing distribution of local minima from the global minimum

achieve global descent. A genetic algorithm would be an example of such a technique. It is worth noticing that genetic algorithms have been previously applied successfully to Multiple Gravity Assist problems.

2.4.4 Multi-objective optimisation

It is apparent from the analysis of the C3 optimal minima that a single objective optimisation is flawed as it allows good solutions to be overlooked that are slightly less optimal in the single objective but would be considered superior overall. Therefore it is logical to use a multi-objective optimisation in preference to a single objective optimisation.

C3 and mission time

One useful multi-objective optimisation problem is a simultaneous minimisation of C3 and mission time. By optimising C3 alone it is easy to ignore trajectories that require marginally more thrust yet significantly reduce mission time. Rather than grid sample, as for the Earth-Mars transfer, it was found to be more effective to initialise the pareto front algorithm with the minima found by SQP, as these tended to fit the unpowered swingby constraint very well and the large number of minima more densely covered the pareto front. Using this method, it was found that 62 out of the 5000 located minima lay close to the pareto front. Using grid sampling with a density of 50 (125000 function evaluations in all) only 8 such points were located.

Figure 2.14 represents the approximation of the pareto front as blue circles onto a plot of all the identified minima. The C3 optimal solution has also been added as a red star and labelled to confirm that it lies on the pareto front. The front is divided into three discrete sections, where each section is spread over one of the t_0 quasi-periodic manifolds - this division of the pareto front is attributable to the unpowered swingby constraint.

2.4.5 Discussion

The EJS transfer is the simplest Gravity Assist trajectory, and therefore the classification of its optimisational complexity certainly does not apply to all Multiple Gravity Assist trajectories. This is because as more gravity assists are considered the problem complexity will obviously increase due to the curse of dimensionality, particularly when the sequence of swingby planets is also allowed to vary, thus introducing a combinatorial elements into the problem. Also, by relaxing the constraint of only unpowered swingbys many more minima will be introduced into the search space. This would likely

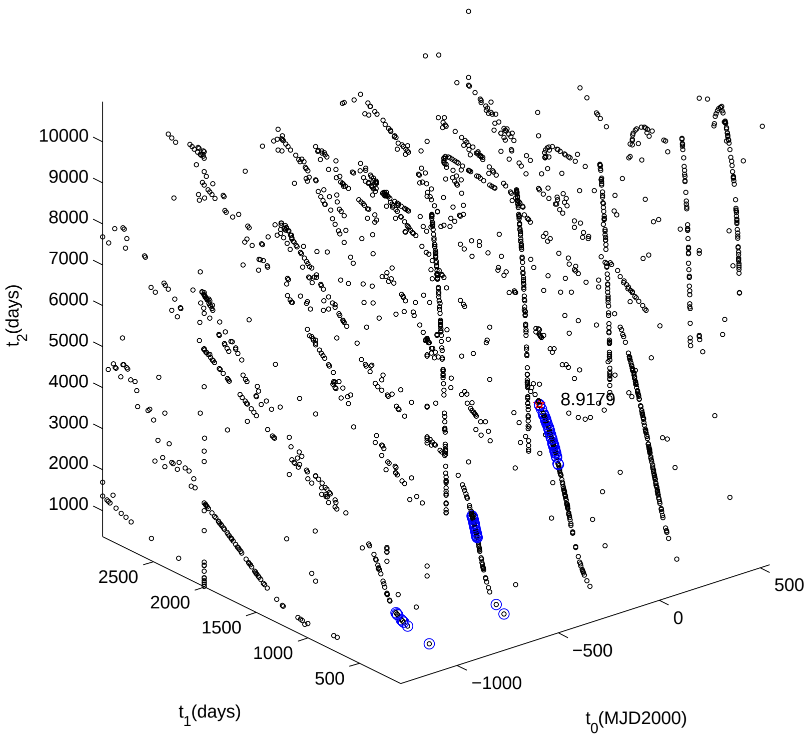

Figure 2.14: An approximation of the Pareto front of the EJS Gravity Assist problem

lead to the creation of valleys rather than many point optima, and therefore the embeddedness of the global minimum would be much greater, increasing its effective basin of attraction to that of the entire valley. However, the complexity of optimisation will rise with the number of swingby planets.

When solving joint combinatorial and continuous optimisation it is more intuitive to use global techniques such as genetic algorithms as they can immediately be applied to such hybrid domains. An alternative approach is to consider the combinatorial and continuous aspects separately by selecting combinations of swingby planets and then optimising the remaining continuous function using an appropriate local method. The former may suffer from the problem of ignoring most of the continuous gradient information within the search space, whereas the latter can be very computationally expensive. Therefore, some compromise is desirable that can effectively exploit gradients within continuous dimensions while meaningfully exploring combinatorial elements.

2.5 Low Thrust Arcs

The third relevant mission design problem to be considered is that of low thrust arcs. This problem is of interest as probes with low thrust propulsion have much higher fuel efficiency than standard chemical thrusters, or in the case of solar sails require no propellant at all for primary propulsion [14]. Consequently, such propulsion systems are suitable for long-term exploration of the outer planets within the solar system, especially when combined with gravity assist manoeuvres.

The test case for this problem is a low thrust Earth-Mars transfer. The probe is assumed to escape the Earth's gravitational influence with the aid of a launcher, and then exerts a constant thrust throughout the mission where the control trajectories of the thrust angle must be optimised.

2.5.1 Classification of Problem Type

This problem can be classified within the taxonomy developed during Work Package 1 as follows:

Variable	**U**	**#**	**B**	$f(t)$	**O**
$\mathbf{x}_i$				X	N/A
$\dot{\mathbf{x}}_i$				X	N/A
t_i			X	N/A	N/A
$\mathbf{x}_f$				X	
$\dot{\mathbf{x}}_f$				X	
t_f			X	N/A	N/A
Additional Constraints					

Control Class	**Subclass**	**Check/Value**
Impulsive	Fixed n-Impulsive	
Continuous	Simple	X
Impulsive	Variable Impulsive	
Continuous	Complex	

Variable	**Check**
Thrust	X
Mission time	
Velocity	
Angular momentum	
Other variables	

Spacecraft Model	**Check**
Point-mass	X
Rigid body	
Low level control	

Astrodynamic Model	**Check/Value**
n-body dynamics (2+)	2
Non-planar model	X
Elliptic orbits	X
Other factors	

The control trajectories are defined, in spherical co-ordinates, by the functions of time $\theta(t)$ and $\phi(t)$. The thrust vector, $\vec{\zeta}$, at time t is therefore

$$\vec{\zeta} = \frac{T}{m - \dot{m}t} \begin{pmatrix} \cos\theta(t)\sin\phi(t) \\ \sin\theta(t)\sin\phi(t) \\ \cos\phi(t) \end{pmatrix} \tag{2.5.1}$$

relative to the spacecraft's current direction of travel, where $\dot{m}$ is the rate of decrease of mass due to propellant flow. This infinite dimensional optimal control problem is solved by approximating the functions $\theta(t)$ and $\phi(t)$ using B-Splines.

2.5.2 B-Splines

As with all continuous control problems, some technique must be used to reduce the infinite-dimensional control problem to a finite-dimensional one [5]. An effective way of doing this is through the use of piecewise continuous polynomials known as *splines*. One convenient form that can be utilised is that of B-splines [49], which are a generalisation of bezier curves. A B-spline is defined by a set of p control points, a knot vector and an order r.

A B-Spline is defined as

$$y(t) = \sum_{j=1}^{p} B(j, r, t)\mathbf{c}_j, \tag{2.5.2}$$

where $\mathbf{c}_j$ is the j^{th} control point and $B(j, r, t)$ is the blending function.

The Cox-de Boor formula is used to calculate the blending coefficients. This formula is recursive, and therefore is split into a base case and a general case. The base case is very simple:

$$B(j, 1, t) = \begin{cases} 1, & \text{if } t_j \le t < t_{j+1}; \\ 0, & \text{otherwise.} \end{cases} \tag{2.5.3}$$

For $r > 1$ the formula recurses, each recursion requiring the evaluation of two blending functions of lower order, hence requiring 2^r evaluations overall.

$$B(j, r, t) = \frac{t - t_j}{t_{j+r-1} - t_j} B(j, r-1, t) + \frac{t_{j+r} - t}{t_{j+r} - t_{j_1}} B(j+1, r-1, t) \qquad (2.5.4)$$

Due to the interpreted nature of MATLAB, using higher order splines is very computationally expensive and hence only second order splines will be used for this investigation, although with subsequent implementations in C++ this will be feasible.

2.5.3 Decision Variables

The decision variables to be optimised were

- t_0, the launch date (in MJD2000);
- t, the mission time (in days);
- T, the thrust of the probe (in Newtons);
- v_0, the hyperbolic excess velocity from Earth (in km/s);
- 6 B-Spline control points $\alpha_{0..5}$ to define $\theta(t)$ (in radians);
- 6 B-Spline control points $\beta_{0..5}$ to define $\phi(t)$ (in radians).

The acceptable levels of thrust and specific impulse were selected to be feasible using xenon ion propulsion systems, for example the Boeing 702 thruster [9]. The specific impulse of the fuel was assumed to be 3000s, and the thrust level between 50mN and 200mN. Two main simplifying assumptions are made here: the thrust is constant over the trajectory and the hyperbolic excess velocity is constrained to be parallel to the Earth's velocity at launch. The knots within the B-Splines were spaced uniformly throughout the mission time. The overall optimisation problem is therefore 16 dimensional.

The bounds on the variables were as follows:

- $t_0 \in [800, 3800]$
- $t \in [150, 500]$
- $T \in [0.05, 0.2]$
- $v_i \in [2.5, 3.0]$
- $\alpha_{0..5} \in [-pi, pi]$
- $\beta_{0..5} \in [-\frac{pi}{2}, \frac{pi}{2}]$

The search space was normalised to the unit hypercube in order to aid convergence of SQP. However, it is worth noting that if a much larger launch window (ten times or above) were considered such a normalisation would impede convergence as the objective function would be immensely oscillatory at that scale. With bounds covering only 4 synodic periods this is not an issue.

2.5.4 Objective function

Although it is usual when using SQP to utilise a multiple shooting method and add equality constraints at each breakpoint [18], this methodology is not practical when considering global optimisation algorithms, since the number of decision variables increases significantly compared with a single shooting method. Therefore, a single shooting method with a scalar objective function was developed, since this is more amenable to global optimisation algorithms. For the numerical integration, 100 equally sized steps of fourth-order Rünge-Kutta [38] will be used. Although using a variable step length procedure would increase accuracy, using the fixed step size allows a simple method of including the thrust control as the thrust direction can be assumed to be constant over a small time period.

Overall, this problem can be seen as a generalisation of the low thrust arc considered in [47] with the addition of non-zero C3.

Rather than using a linear combination of terms as an objective function, a quadratic function was selected in order to increase the convexity of the objective function. The constraint on reaching Mars was enforced by a penalty term, and therefore the objective function was of the form

$$f = a \cdot v_f^2 + b \cdot m_p^2 + c \cdot r_f^2. \tag{2.5.5}$$

where v_f is the final velocity relative to Mars at the end of the trajectory (m/s), m_p is the total mass of propellant required (kg) and r_f is the distance from Mars at the end of the trajectory (km). The weights were selected using heuristic considerations as $a = \frac{1}{140}$, $b = \frac{1}{500}$ and $c = \frac{1}{r_s}$, where r_s is the radius of Mars' sphere of influence ($1.08 \times 10^6 km$). Although it would seem advantageous to place a larger weight on the positional discrepancy, it was found that using such weighting led to reaching Mars reliably but with a large relative velocity. Subsequently, it was discovered that by significantly increasing the importance of optimising the velocity, such that the importance of both positional and velocity discrepancy would be approximately equal initially, much better solutions were located.

2.5.5 Results

SQP was run to a terminal number of 200 iterations and the resulting minima was recorded. A total of 180 minima were obtained overall. When the minima were grouped, using a relatively large minimum Euclidean distance of 0.25 (after normalisation of the decision vectors), 176 unique minima were found.

Table 2.6 shows the best forty minima in terms of objective score f, although it can be seen that this is almost identical to ordering by increasing propellant mass. This is due to the design of the objective function, as once a trajectory with a small positional and velocity discrepancy has been found the mass term becomes the most significant.

Although the lower bound on the mission time is 150 days, it is evident that the best minima have relatively long mission times at greater than 300 days. Figure 2.15 shows the launch date and mission time of all the optima - as expected there is a quasi-periodicity with respect to t_0 and the minima correspond well to the valleys observed in the bi-impulsive Earth-Mars transfer objective function (Figure 2.2). The four best minima will now be presented in decreasing order of quality.

Table 2.6: The forty best minima located by SQP in the low thrust arc problem

Minimum	t_0(MJD2000)	t(days)	f	r_f(km)	v_f(m/s)	m_p(kg)
1	2796.1903	469.6504	0.045838	10876.3428	7.1599	105.967
2	2040.2173	397.4957	0.046638	1109.2421	1.0653	107.9572
3	2038.1235	398.189	0.047629	4340.8604	3.1651	108.9179
4	2038.355	390.0254	0.050538	6715.2394	1.2911	112.3305
5	2038.4939	362.408	0.050623	1887.9589	1.8789	112.4315
6	2808.2024	500	0.051422	6014.5115	3.1503	113.173
7	2038.045	351.6834	0.05231	3498.444	1.7322	114.2932
8	2032.8415	408.4244	0.054447	1373.0241	0.8913	116.6545
9	2796.2383	440.5307	0.054688	6515.2971	14.9337	113.0081
10	2801.7847	447.7984	0.054732	1539.4369	1.3083	116.9429
11	2035.3783	357.9382	0.055346	2951.8443	1.3385	117.5904
12	2801.0424	420.6035	0.055486	1062.7311	0.65268	117.7696
13	2032.0505	384.02	0.056146	1531.2477	1.1432	118.4515
14	3552.833	485.0083	0.056432	7566.8566	2.7957	118.5937
15	2803.7189	439.2496	0.057121	1351.7074	1.1334	119.4767
16	2035.2439	359.1207	0.05733	2058.113	1.4726	119.6788
17	2033.6303	374.3617	0.057454	3688.8252	2.5623	119.7257
18	2793.8601	457.9182	0.060124	1705.0256	1.174	122.576
19	1257.8589	337.2365	0.060223	2336.8227	1.2326	122.6726
20	2041.1421	431.6797	0.060673	1469.6377	1.3396	123.1284
21	2038.8212	412.1151	0.061181	38709.9541	7.4229	121.4648
22	2799.3874	414.6924	0.061396	5548.4891	1.3028	123.8367
23	2033.3507	402.7306	0.061945	5057.9882	3.1346	124.2637
24	1252.506	339.7791	0.062111	1562.8953	1.1744	124.5857
25	2030.1049	356.8136	0.062349	3394.796	1.0516	124.8212
26	2806.2716	425.9285	0.062609	9391.8813	3.7412	124.8091
27	2031.0278	375.8207	0.06279	1909.8359	1.2131	125.2626
28	2036.6804	444.5512	0.06315	1067.7365	0.88478	125.6346
29	2800.7225	427.3024	0.063477	2282.0005	1.6818	125.9238
30	2042.0018	350.4631	0.06428	6862.8627	3.708	126.5102
31	1247.5734	320.8222	0.064799	6609.0642	2.029	127.177
32	2038.7419	397.796	0.064923	1258.9612	0.8616	127.3866
33	1253.6328	335.1578	0.066144	8257.887	3.2116	128.3752
34	1255.0665	368.596	0.066232	2051.0805	1.0155	128.6582
35	1248.4221	302.1041	0.066271	1954.8465	1.2188	128.6897
36	3559.5261	477.9605	0.067779	2295.5367	1.1005	130.1487
37	1256.1459	318.007	0.068089	7183.5941	3.4362	130.2454
38	2033.0183	346.1051	0.068157	2097.5758	1.125	130.5115
39	1242.5572	315.849	0.070938	2639.5153	1.2889	133.1399
40	2043.3409	388.1688	0.070999	1262.8674	0.84742	133.216

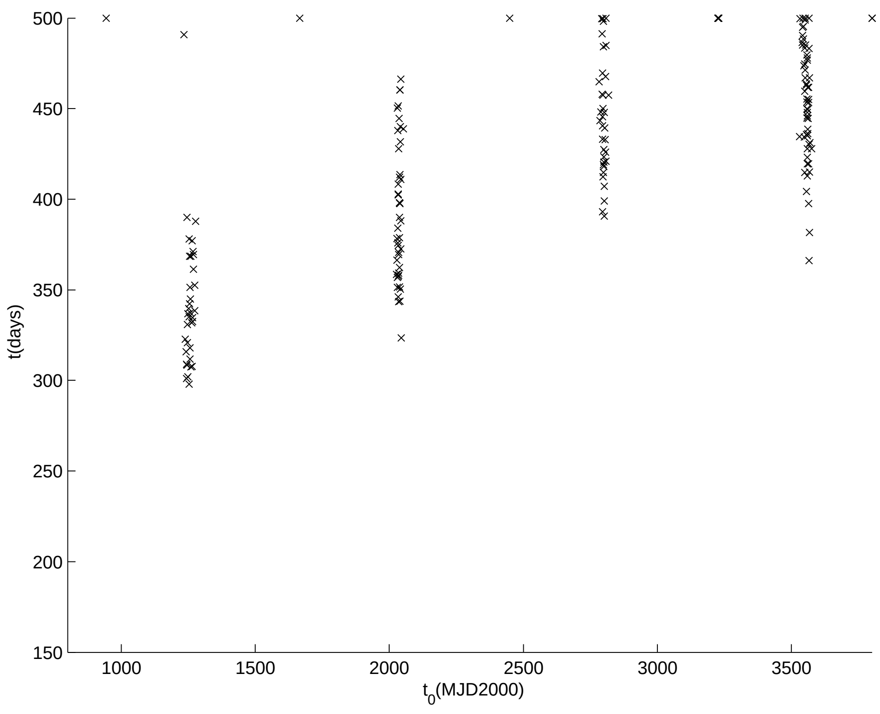

Figure 2.15: A plot of launch date vs mission time to illustrate the quasi-periodicity in t_0

Evaluation of minimum 1

Minimum 1 is the best solution found, both in terms of the requisite propellant mass and the overall objective function value (0.045838). The values of the decision variables and objectives for the best solution follow in tabular form:

Decision variables			
t_0(MJD2000)	2796.1903	t(days)	469.6504
T(N)	0.076855	v_i(km/s)	2.693
θ_0(rad)	0.39363	θ_1(rad)	0.3253
θ_2(rad)	0.57601	θ_3(rad)	0.39823
θ_4(rad)	0.1728	θ_5(rad)	0.022781
ϕ_0(rad)	-0.76365	ϕ_1(rad)	0.78749
ϕ_2(rad)	-0.080502	ϕ_3(rad)	-0.16416
ϕ_4(rad)	-0.48796	ϕ_5(rad)	-0.70066
Objectives			
f	0.045838	r_f(km)	10876.3428
v_f(m/s)	7.1599	m_p(kg)	105.967

The resulting trajectory is shown in Figure 2.16 and the corresponding control trajectories in Figure 2.17. The blue line plotted at each integration step indicates the orientation of the thruster. The mission time is close to the top bound of 400 days, and the transfer angle is large at over 4.5 radians - this is compared to the optimal bi-impulsive trajectory located in section 3.5 which had a transfer angle of significantly less than π radians.

Due to the small number of samples (180) and the high dimensionality of this problem, it is likely that this solution is not the true global optimum. However, by considering this solution as a test case, conclusions may still be drawn about the complexity of finding good solutions and it follows that this forms an upper bound on the true global optimum.

Evaluation of minimum 2

Minimum 2 is both the second best solution by propellant as well as by objective function value. The associated decision variable values and objectives are shown in tabular form:

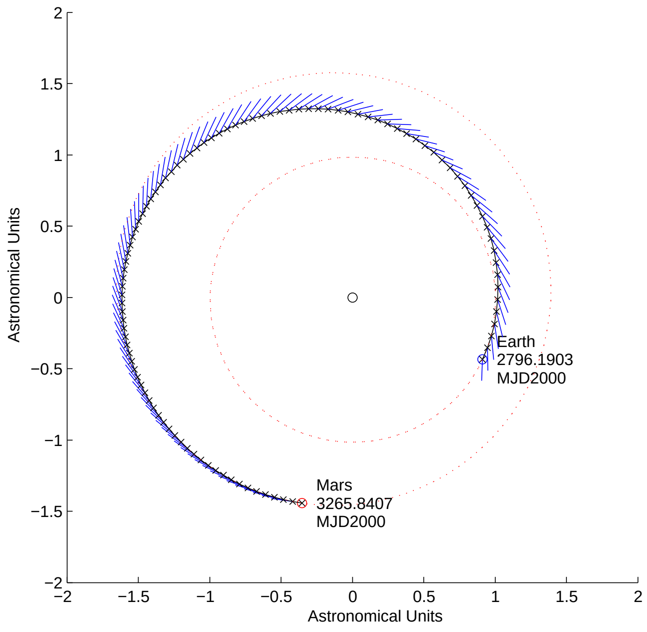

Figure 2.16: The resulting low thrust arc for minimum 1

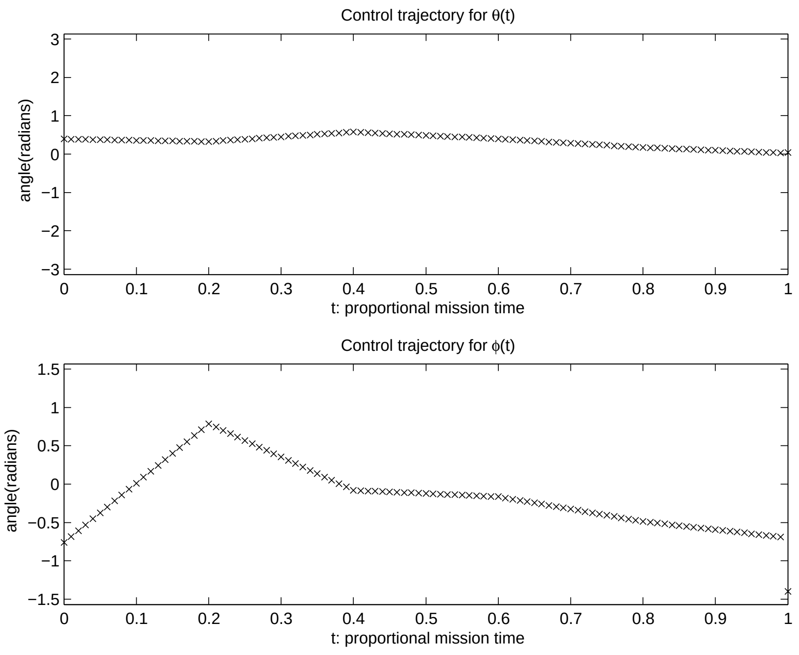

Figure 2.17: The optimised control trajectories for $\theta(t)$ and $\phi(t)$ for minimum 1

Decision variables			
t_0(MJD2000)	2040.2173	t(days)	397.4957
T(N)	0.092511	v_i(km/s)	2.8682
θ_0(rad)	-0.42454	θ_1(rad)	0.013761
θ_2(rad)	0.45645	θ_3(rad)	0.20598
θ_4(rad)	0.58845	θ_5(rad)	-1.3898
ϕ_0(rad)	-1.1732	ϕ_1(rad)	1.258
ϕ_2(rad)	0.45972	ϕ_3(rad)	0.17527
ϕ_4(rad)	-0.40909	ϕ_5(rad)	-0.0024878
Objectives			
f	0.046638	r_f(km)	1109.2421
v_f(m/s)	1.0653	m_p(kg)	107.9572

The corresponding transfer trajectory and control trajectories can be observed in Figure 2.18 and 2.19. The launch date is approximately two years later than minimum 1, but the mission is significantly shorter at 397 days compared to 469 days.

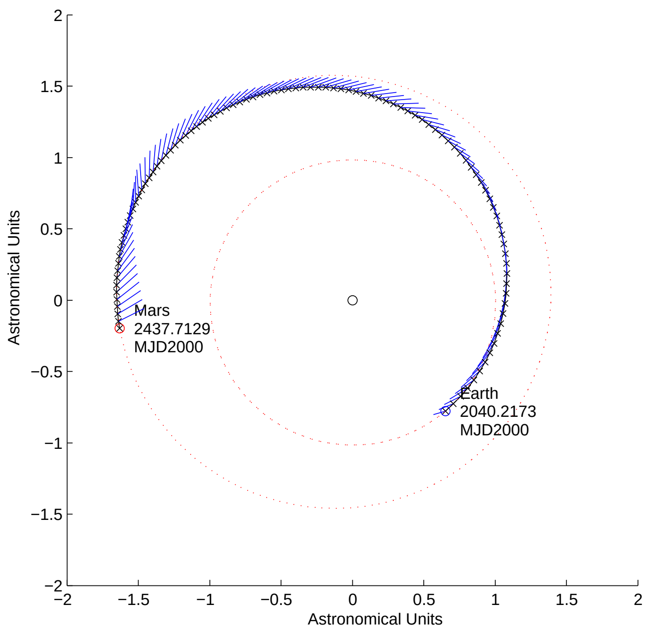

Figure 2.18: The resulting low thrust arc for minimum 2

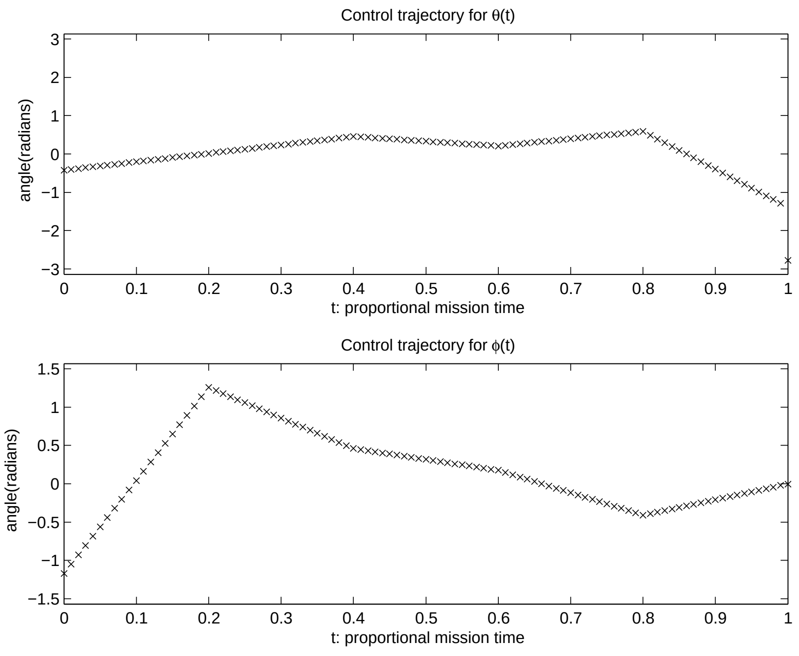

Figure 2.19: The optimised control trajectories for $\theta(t)$ and $\phi(t)$ for minimum 2

Evaluation of minimum 3

The values of the decision variables and objectives for the 3rd best solution follow in tabular form:

Decision variables			
t_0(MJD2000)	2038.1235	t(days)	398.189
T(N)	0.093172	v_i(km/s)	2.9603
θ_0(rad)	0.19127	θ_1(rad)	0.72799
θ_2(rad)	0.59077	θ_3(rad)	0.92772
θ_4(rad)	0.028092	θ_5(rad)	-0.99898
ϕ_0(rad)	0.91057	ϕ_1(rad)	0.35657
ϕ_2(rad)	0.41216	ϕ_3(rad)	0.71962
ϕ_4(rad)	-0.26583	ϕ_5(rad)	-0.57385
Objectives			
f	0.047629	r_f(km)	4340.8604
v_f(m/s)	3.1651	m_p(kg)	108.9179

The low thrust arc is displayed in Figure 2.20 and the control trajectories in Figure 2.21. The trajectory and launch date/mission time are very similar to Minimum 2, although the control trajectories can be seen to vary significantly.

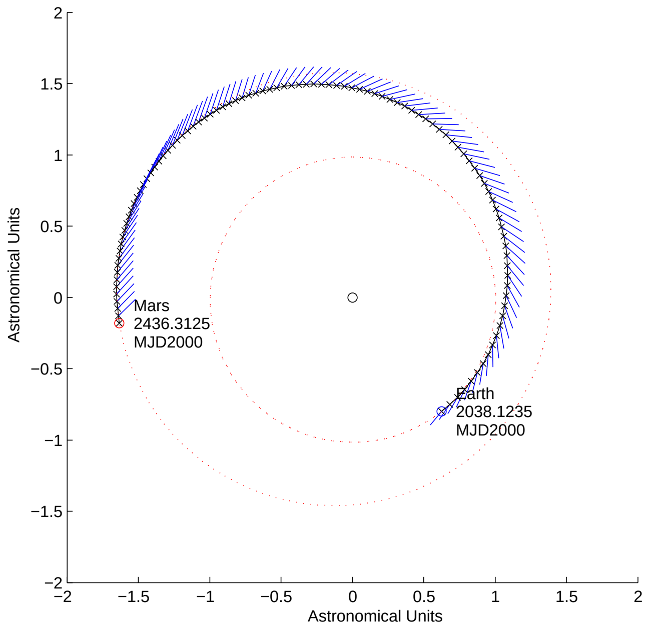

Figure 2.20: The resulting low thrust arc for minimum 3

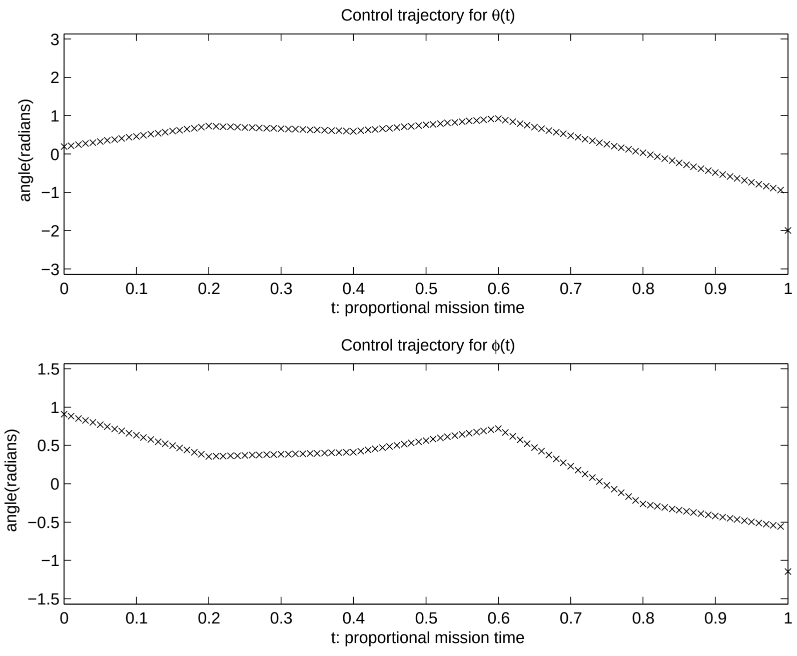

Figure 2.21: The optimised control trajectories for $\theta(t)$ and $\phi(t)$ for minimum 3

Evaluation of minimum 4

The values of the decision variables and objectives for minimum 4 follow in tabular form:

Decision variables			
t_0(MJD2000)	2038.355	t(days)	390.0254
T(N)	0.098103	v_i(km/s)	2.8647
θ_0(rad)	-1.9858	θ_1(rad)	0.35929
θ_2(rad)	0.12126	θ_3(rad)	-0.15128
θ_4(rad)	0.85795	θ_5(rad)	-0.95531
ϕ_0(rad)	-0.24717	ϕ_1(rad)	0.83485
ϕ_2(rad)	0.25799	ϕ_3(rad)	0.69296
ϕ_4(rad)	-0.53301	ϕ_5(rad)	-0.054793
Objectives			
f	0.050538	r_f(km)	6715.2394
v_f(m/s)	1.2911	m_p(kg)	112.3305

The corresponding trajectory and control trajectories are shown in Figure 2.22 and 2.23, respectively. It is evident that this solution is similar to minima 2 and 3 in terms of launch date/mission time, although again significantly different control trajectories are observed. It appears that this family of solutions are close to optimal, and since minimum 1 requires only 2kg less propellant than minimum 2 yet has mission time of over two months longer, in practical terms minimum 2 can be considered superior.

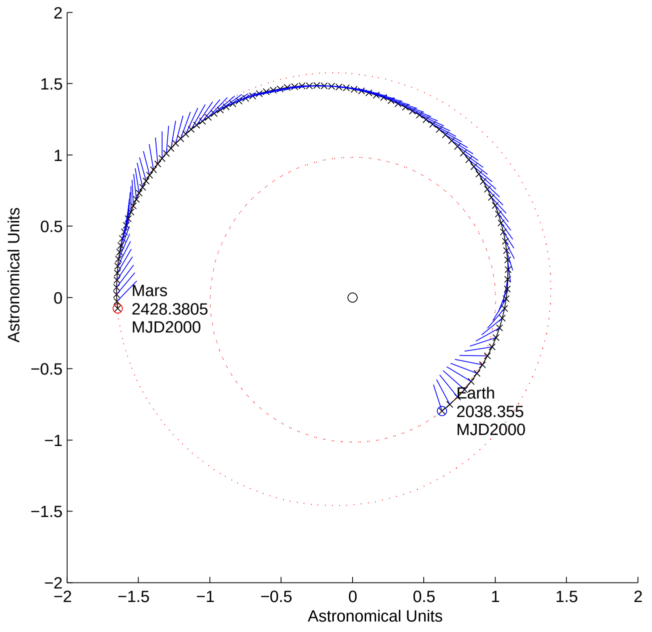

Figure 2.22: The resulting low thrust arc for minimum 4

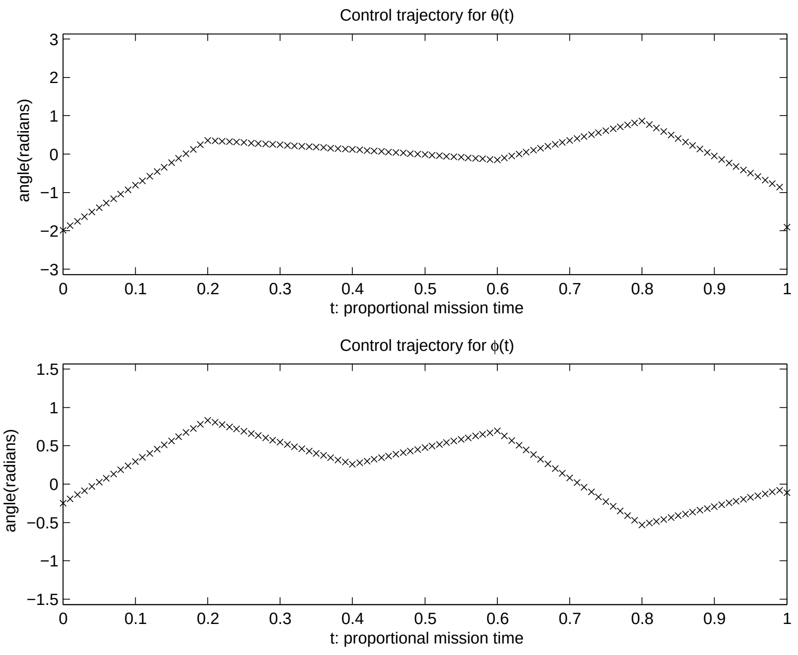

Figure 2.23: The optimised control trajectories for $\theta(t)$ and $\phi(t)$ for minimum 4

2.5.6 Size of the basin attraction of the global minimum

Although the best solution located empirically is probably not the true global optimum, it is most likely that the size of the basin of attraction of the true global optimum is smaller than that of the best optimum found, which allows qualitative conclusions to be drawn. Therefore, using an unbiased estimator the upper bound for $\hat{p}^*$ can be determined by

$$\hat{p}^* < \frac{1}{d+1}, \tag{2.5.6}$$

as the best solution was located only once out of d trials. For 180 trials, this yields $\hat{p}^* < 0.0056$.

A basin of attraction that occupies less than 1% of the search space means that $(1 - \hat{p}^*)$, the probability of not sampling within the basin of attraction, may be classified as *large*.

2.5.7 Number of function evaluations

The acceptable number of function evaluations, N_f, is much lower for low thrust arcs than the problems investigated in Sections 3.5 and 2.4. This is because the numerical integration required to propagate the trajectory is much more computationally expensive than solving Lambert problems.

It is also worth noting that the hypervolume of the 16D search space is incredibly large and consequently very difficult to analyse effectively. For example, consider grid sampling the objective function (as implemented on the Earth-Mars transfer and Gravity Assist trajectories) - even with a quantisation of only three samples in each dimension forty-three million objective function evaluations would be required.

2.5.8 Number of local minima

The number of unique local minima is determined by grouping together minima with a Euclidean distance between their decision vector smaller than some tolerance, ϵ. However, due to the vastly differing bounds of the search space, it is necessary to normalise this search space to the unit hypercube in order to find identical minima, otherwise any tolerance used will be meaningless. It was found that even using a relatively high tolerance of 0.25 (when the hyper-diagonal of the space is 4), only four of the minima are classed as identical to another minimum. Therefore, the number of local minima must be much higher than one hundred and eighty in order to observe such a small overlap.

2.5.9 Embeddedness of best solution

Figure 2.24 shows a histogram illustrating the distribution of the number of local minima with the Euclidean distance from the best solution found. It can be seen that the best solution is isolated as there are no local minima within a normalised distance of 0.1 from it. As a comparison, Figure 2.25 shows similar diagrams using four randomly selected minima instead of the best solution: the graphs are visually very similar to that in Figure 2.24. Also, for any of these five minima used as a centre point there are no minima located at a distance greater than 0.5. It is hypothesised that the minima are scattered approximately uniformly within the search space - this is supported by the fact that the hypervolume of a 16D incremental hypersphere with radius in the interval $[r, r + \delta r]$ is proportional to r^{16}, hence for low r the hypervolume is negligible. This explains why no minima are found at these distances, but then the count increases rapidly with the radius, followed by a decrease. This results in the mode observed on the histograms. Notice that the decrease observed in the histograms is explained by the intersection of the incremental hypersphere with the boundary of the search space, which is a hypercube. As r increases, the hypervolume of this intersection decreases.

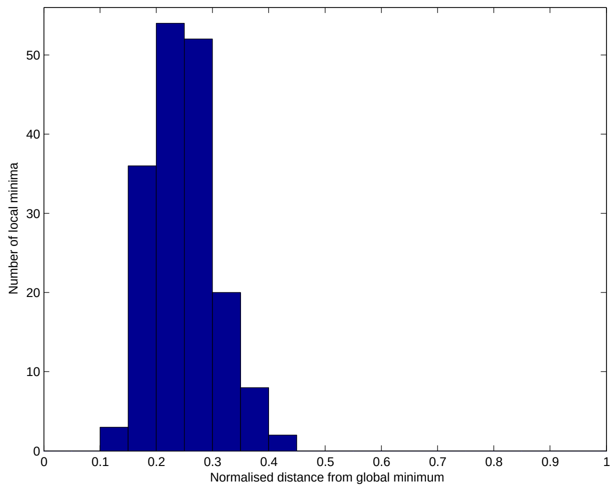

Figure 2.24: A plot of the number of local minima as a function of distance from the global minimum

2.5.10 Classification of problem complexity

The low thrust arc problem can therefore be classified as follows:

$(1-\hat{p}^*)^{N_f}$	Large
Embeddedness	Isolated
#mins	High
Classification	D_2 **- difficult**
Recommended technique	**Global optimiser**

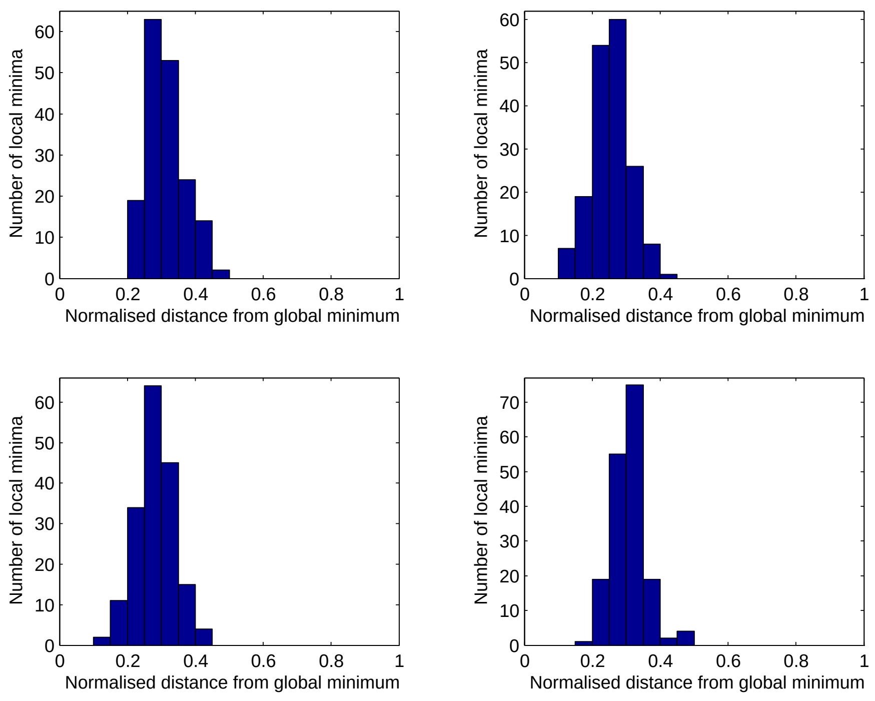

Figure 2.25: Plots of proximity of other local minima from randomly selected minima

2.5.11 Discussion

Although for this experiment numerical propagation was used in order to integrate trajectories, future work will investigate the use of exponential sinusoids [35], as these remove the need for numerical propagation and are thus more computationally efficient [31].

2.6 Conclusions

This report has described some relevant mission design problems and characterised their complexity through empirical investigation of four main properties: the size of the basin of attraction for the global optimum, the number of acceptable function evaluations, the embeddedness of the global optimum and the number of local minima. By performing such a classification, as discussed by Törn, appropriate methods for their effective solution may be selected. The mission design problems that have been studied were bi-impulsive interplanetary transfers, Multiple Gravity Assist manoeuvres and interplanetary low thrust arcs.

Future work will consider appropriate global techniques that can be used to solve such problems, based on the analysis of the properties of the search space presented here, and then proceed to develop and analyse novel techniques for this purpose. It is perceived that by approaching each problem individually and incorporating domain knowledge into novel optimisation algorithms the efficacy of their solution may be increased. Ideally, such algorithms would contain both effective global and local aspects in order to obtain an optimal ratio of exploration and exploitation.

Chapter 3

Selection of Appropriate Global Algorithms

This report discusses the selection of appropriate global algorithms to optimise the relevant mission design problems of interplanetary transfers, and multiple gravity assist trajectories (both with and without deep space manoeuvres). Both deterministic and stochastic algorithms are considered and tested on a series of benchmarks for standardised comparison before being applied to the selected actual mission design problems. It is shown that modern global optimisation methods perform significantly better on mission design problems than standard genetic algorithms.

3.1 Introduction

Work Package 2 provided an analysis of the complexity of three relevant mission design problems: impulsive thrust optimal interplanetary transfers, gravity assist trajectories and low thrust interplanetary transfers.

The purpose of Work Package 3 is to select appropriate global optimisation algorithms to optimise relevant mission design problems and assess their performance. The focus here will be on multiple gravity assist trajectories, although interplanetary transfers will also be investigated. The manually integrated low thrust transfer example used in the previous work package was not considered as it does not represent the state of the art for optimising low thrust transfers: exponential sinusoids [35] provide a more computationally efficient way to optimise such transfers, and so may be investigated in further work in the context of adding low thrust arcs to multiple gravity assist trajectories.

Section 2 describes the classification of global algorithms, based on the reliability with which they reach the global minimum. Section 3 then presents a selection of modern optimisation techniques including deterministic, model-based and instance-based stochastic methods that have performed well on standardised benchmarks. Section 4 proceeds to compare all these algorithms directly on a custom set of benchmarks, and sections 5 and 6 on the desired mission design problems.

3.1.1 Notation

The following notation will be used throughout this report :

f^* a minimum value of an objective function f;

$\hat{f}^*$ an estimate of f^*;

$\mathbf{x}$ a vector in $\mathbb{R}^n$;

$\dot{\mathbf{x}}$ the derivative of $\mathbf{x}$ with respect to time.

3.2 Classification of Global Algorithms

Neumaier [33] describes four classes of global optimisation algorithms. These are incomplete, asymptotically complete, complete and rigourous methods in ascending order of robustness.

Incomplete Incomplete methods use heuristics for searching but have no safeguards if the search gets stuck in a local minima

Asymptotically Complete Asymptotically complete methods will reach a global minimum if allowed to run indefinitely, but have no means of determining whether a global optimum has been found

Complete Complete methods reach a global minimum after indefinite runtime, and know after a finite time that a global minimum has been found to a prescribed tolerance

Rigourous Rigourous methods are complete methods that reach a global minimum even in the presence of rounding errors, except in near-degenerate cases where the tolerances may be exceeded

From the above description, it initially appears that all incomplete methods should be discarded in deference to asymptotically complete and better methods, but this is not necessarily so. Although incomplete methods cannot *guarantee* a global minimum, due to the fact that most such methods are stochastic, they *can* guarantee a high probability of locating it in a feasible number of function evaluations.

Conversely, complete methods can guarantee the location of the global minimum (to a tolerance) after a finite time, but this is generally only after a number of function evaluations rising exponentially with problem dimensionality - on most real optimisation problems this is not practical. Hence, some heuristic stopping criteria must be applied, yielding an *incomplete* search. For example, a grid search at a desired resolution is a complete method, as the solution is definitely reached after a finite time, but it would certainly not be a good choice for accurate high-dimensional continuous global optimisation due to the vast number of function evaluations required.

Therefore, the fact that stochastic algorithms such as Genetic Algorithms and Particle Swarm Optimisation are incomplete in nature should not discourage their application to black box objective functions: in all realistic cases, *all* applied algorithms will similarly produce an incomplete search.

3.3 Global Algorithms

This section describes the different global algorithms to be applied to mission design problems. The selection includes deterministic algorithms, and both model-based and non-model-based probabilistic algorithms.

3.3.1 Deterministic Algorithms

The two deterministic algorithms that have been considered are both based on the branch and bound methodology, although Multi-level Coordinate Search (MCS) also makes use of local search to increase efficiency.

DIRECT - Divided Rectangles

DIRECT, [25], is a complete deterministic global optimiser for bound constrained optimisation. The simple principle allows Lipschitzian optimisation of a given objective function without *a priori* knowledge of the Lipschitz constant. To achieve this, DIRECT samples the centre of the set of hyper-rectangles $\mathcal{S}$, and then subdivides those that lie on the convex hull of hyper-diagonal length to objective function value (see Figure 3.1). In infinite time, this method performs an exhaustive search of the entire search space. The following algorithm description is taken from Baker, 2001. [1].

```
DIRECT algorithm
1. Normalize the search space to be the unit hypercube.
   Let $c_1$ be the centerpoint of this hypercube and evaluate $f(c_1)$.
2. Repeat
3.     Identify the set $\mathcal{S}$ of potentially optimal rectangles (those
       rectangles defining the bottom of the convex hull of a scatter
       plot of rectangle diameter versus $f(c_i)$ for all rectangle
       centres $c_i$).
4.     For all rectangles $j \in S$:
5.         Identify the set $\mathcal{I}$ of dimensions with the maximum side
           length. Let $d$ equal one-third of this maximum side length.
6.         Sample the function at the points $c \pm \delta e_i$ for all $i \in \mathcal{I}$,
           where $c$ is the centre of the rectangle and $e_i$ is the $i^{th}$ unit
           vector.
7.         Divide the rectangle containing c into thirds along the
           dimensions in $\mathcal{I}$, starting with the dimension with the
           lowest value of $f(c \pm \delta e_i)$ and continuing to the dimension
           with the highest $f(c \pm \delta e_i)$.
8.     end
9. Until Convergence
```

MCS - Multilevel Coordinate Search

MCS [23] is a complete deterministic global optimiser that uses branching and local search. It was based on the principles encapsulated by the DIRECT algorithm but applies a more flexible branching scheme and incorporates local search to more effectively estimate the lower bounds of sub-

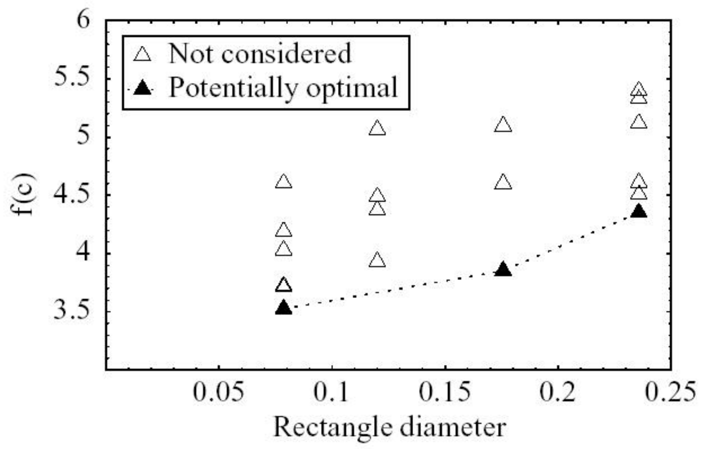

Figure 3.1: Diagram illustrating the subdomain selection process of DIRECT(reproduced from Baker, 2001)

domains. Although MCS has many integrated heuristics and lacks the elegant simplicity of DIRECT, it has proven to be a very effective global optimiser over a range of benchmarks [22]. The complexity of the algorithm prohibits a summary, and for more information readers are referred to the original paper [23].

3.3.2 Model-Based Stochastic Algorithms

Model based algorithms use some probabilistic model of the parameter space in order to bias the sampling of the search space towards good points, and therefore fall under the umbrella term of Estimation of Density algorithms [32]. The probabilistic model is then updated based on the results of the current sample. The simplest methods assume a unimodal distribution for each parameter i.e. all parameters are independent, although more complex approaches use Bayesian techniques to infer relationships between different variables.

PGSL - Probabilistic Global Search Lausanne

PGSL [39] is an incomplete probabilistic model based global optimiser. PGSL differs from most other global optimisation algorithms in that it represents the probability distribution as a histogram rather than in a parametric form. This has the advantage that PGSL can handle arbitrary multi-modal distributions, unlike others, such as Cross Entropy, which use Gaussian models.

The PGSL algorithm uses four embedded loops - the iteration cycle, focusing cycle, probability updating cycle and sampling cycle.

```
PGSL: Main algorithm
1.  Choose complete domain $\mathcal{A}$ as the current sub-domain. Set
    best domain, $\mathcal{B}$, to NULL.
2.  Repeat
3.      Complete $N_{fc}$ iterations of the focusing cycle: select
        the subdomain with best solution $\mathcal{A}^\star$.
4.      If $f(\mathcal{A}^\star) < f(\mathcal{B})$ then update $\mathcal{B}$.
5.      Choose a smaller sub-domain centred around $\mathcal{B}$ as the
        current sub-domain.
6.  Until Convergence.
```

```
PGSL: Focusing Cycle
1.  Assume a uniform PDF throughout current subdomain, set
    SUBDOMAIN-BEST to NULL.
2.  Repeat
3.      Complete $N_{PUC}$ iterations of the probability updating
        cycle; select the best solution, PUC-BEST.
4.      If $f(PUC - BEST) < f(SUBDOMAIN - BEST)$,
        Update SUBDOMAIN-BEST
5.      Subdivide the interval containing the PUC-BEST and
        redistribute probabilities according to an exponential
        decay function.
6.  Until Terminal Iterations.
```

```
PGSL: Probability Updating Cycle
1.  Set PUC-BEST to NULL
2.  Repeat
3.      Sampling cycle: Evaluate $N_S$ samples. Select the best as BEST-SAMPLE.
4.      If $f$(BEST-SAMPLE) < $f$(PUC-BEST), Update PUC-BEST
5.      Increment the probability of the interval containing PUC-BEST.
6.  Until Terminal Iterations.
```

CE - Cross Entropy

Cross Entropy, originally developed for combinatorial optimisation [29], uses a Gaussian distribution to progressively bias the sampling of point towards a global optimum. Although the model used is unimodal for each decision variable, CE can also function well under significant relaxation of this assumption.

```
Cross Entropy algorithm
1. Initialise $\mu$ and $\sigma$
2. Repeat
3.         Sample $k$ points from current probability distribution
4.         Select $k\gamma$ samples with best objective function value
5.         Calculate $\mu'$ and $\sigma'$ from the selected samples
6.         Update $\mu = \alpha\mu + (1 - \alpha)\mu'$
7.         Update $\sigma = \alpha\sigma + (1 - \alpha)\sigma'$
8. Until Convergence
```

The standard termination criteria for CE occurs when the standard deviation for each decision variable becomes smaller than some given tolerance.

3.3.3 Instance-Based Stochastic Algorithms

Stochastic algorithms that do not retain some probabilistic models of the search space are instance based.

GA - Genetic Algorithm

The Genetic Algorithm [21] has been frequently applied to mission design problems [6, 47, 48], and therefore is included in this study to provide a direct comparison with the more modern global optimisation techniques. The decision variable vector was encoded as a series of real values, rather than a full binary encoding. This increases the probability of the crossover operation generating better solutions. As a mutation operator a small perturbance was added to a given decision variable: this was generated by a zero mean Gaussian distribution with a standard deviation of 0.03 times the size of the domain for the given decision variable.

```
Genetic Algorithm
1.  Initialise population vectors uniformly over search space.
2.  Repeat
3.          Evaluate objective function for all population members
4.          Select proportion of population $\gamma$ with best objective function values
5.          For each individual in selected set $\mathbf{x}_i$
6.              Select another individual in the set to be 'mated' with
7.              Produce two children through using crossover/mutation operator
8.          end
9.          Replace the less fit members of the population with the newly
            generated children
10. Until Convergence
```

PSO- Particle Swarm Optimisation

PSO [26] was originally designed as a simulation of flocking behaviour in birds, although its potential for optimisation was recognised shortly afterwards. Each particle, analogous to the idea of an individual in genetic algorithms, has a position within the search space and a velocity, both of which are initialised randomly.

As iterations progress, each particle keeps tracks of the position of the best solution it has so far encountered, and also knows the *globally* best solution found by the entire population. The velocity is updated by two main components: the *cognitive* component, which attracts the particle towards its own best solution, and the *social* component, which attracts the particle to the best known solution. The algorithm can be summarised in the following equations:

$$\mathbf{x}_i = \mathbf{x}_i + \mathbf{v}_i \tag{3.3.1}$$

$$\mathbf{v}_i = \omega \mathbf{v}_i + \eta_1 r_1 (\mathbf{x}_p^\star - \mathbf{x}_i) + \eta_2 r_2 (\mathbf{x}_g^\star - \mathbf{x}_i), \tag{3.3.2}$$

where $\mathbf{x}_p^\star$ is the personal best solution of the i^{th} particle, $\mathbf{x}_g^\star$ is the globally best known solution, and r_1, r_2 are uniform random numbers in the interval [0,1].

Particle Swarm Optimisation Algorithm

```
1.  Initialise $\mathbf{x}$ uniformly randomly over search space.
2.  Initialise $\mathbf{v}$ uniformly randomly within hyperparallelipiped
    of scale $v_max$ of the search space.
3.  Repeat
4.      For each population vector $\mathbf{x}_i$
5.          Calculate objective function $f_i$
6.              If $(f_i < f_{pi}^\star)$
7.                  $f_{pi}^\star = f_i$
8.                  $\mathbf{x}_{pi}^\star = \mathbf{x}_i$
9.              end
10.             If $(f_i < f_g^\star)$
11.                 $f_g^\star = f_i$
12.                 $\mathbf{x}_g^\star = \mathbf{x}_i$
13.             end
14.         $\mathbf{v}_i = \omega \mathbf{v}_i + \eta_1 r_1 (\mathbf{x}_p^\star - \mathbf{x}_i) + \eta_2 r_2 (\mathbf{x}_g^\star - \mathbf{x}_i)$
15.         $\mathbf{x}_i = \mathbf{x}_i + \mathbf{v}_i$
16.     end
17. Until Convergence
```

MPSO - Multiple Particle Swarm Optimisation

Multiple Particle Swarm Optimisation (MPSO) is a novel technique based on PSO. It is similar to that investigated by Blackwell [8], which showed significantly improved performance over the basic algorithm. MPSO includes aspects of niching genetic algorithms, in that each swarm evolves separately, for the main part. However, every 3 iterations the swarm membership of two randomly chosen particles is exchanged, effectively introducing new information into those swarms as to the position of other minima. This method has been found in preliminary tests to significantly decrease the probability of PSO getting stuck in a local minimum. Varying the number of swarms manipulates the ratio between exploration and exploitation - only 1 swarm performs a fast optimisation with a significant probability of converging to a local optimum (standard PSO), while a larger number of swarms is more robust but exploits minima less effectively.

DE - Differential Evolution

Differential Evolution [42] is a novel incomplete probabilistic global optimiser based on Genetic Algorithms [21], and was the highest ranked GA-type algorithm in the First International Contest on Evolutionary Computation. Additionally, DE is the standard global optimisation algorithm implemented in *Mathematica* [50]. In their paper, Rainer and Storn [42], considered two crossover schemes, DE1 and DE2. Scheme DE1 will be used as the crossover operator as it was shown to perform the best on the most complex test function they examined, which was an 8 dimensional Chebychev polynomial fitting problem.

Differential Evolution Algorithm (DE1)

```
1.  Initialisation: select population vectors uniformly randomly over
    search space
2.  Repeat
3.      For each population vector $\mathbf{x}_i$
4.          Select two other individuals uniformly randomly over entire
            population, $\mathbf{x}_2$ and $\mathbf{x}_3$
5.          Create test vector $\mathbf{x}'_i = \mathbf{x}_i + F(\mathbf{x}_3 - \mathbf{x}_2)$
6.          Evaluate $f'_i$.
7.          If $(f'_i < f_i)$
8.              Replace $\mathbf{x}_i$ with $\mathbf{x}'_i$
9.          end
10.     end.
11. Until Convergence
```

3.4 Benchmarking Global Optimisation Algorithms

Before focusing on the desired mission design problems, it is useful to consider the performance of these algorithms on standardised benchmark functions. Since such objective functions are much less computationally expensive than mission design problems, they provide both a standardised test and a time efficient way to initially compare algorithms and determine which may be appropriate for a given problem. Also, because the global optima of such test functions are well known this allows validation of the functioning of each global optimisation algorithm and allows comparison with much of the global optimisation literature.

3.4.1 The test function set

Eight different functions were selected in order to assess the performance on multi-modal functions with multiple local minima. This test function set is based on that of Jones *et al* [25], but has had the Rosenbrock and Rastrigin functions added in varying dimensionality to increase the difficulty of the test set.

Branin's function

Branin's test function has 3 global optima.

$$f = h + a(x_2 - bx_1^2 + cx_1 - d)^2 + h(1-e)\cos x_1, \qquad (3.4.1)$$

where $a = 1, b = \frac{5.1}{4\pi^2}, c = \frac{5}{\pi}, d = 6, h = 10$ and $e = \frac{1}{8\pi}$. The boundary constraints for this function are $x_1 \in [-5, 10], x_2 \in [10, 15]$. Figure 3.2 shows the function.

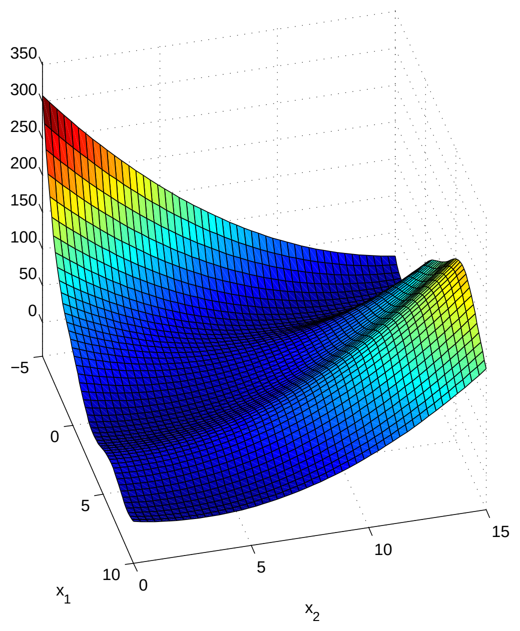

Figure 3.2: The Branin global optimisation test function

Six hump camel function

The six-hump camel function is 2 dimensional and has 2 global minima and 4 local minima [12].

$$f = (4 - 2.1x_1^2 + \frac{x_1^4}{3})x_1^2 + x_1 x_2 + (-4 + 4x_2^2)x_2^2, \tag{3.4.2}$$

with $x_1 \in [-3, 3]$, $x_2 \in [-2, 2]$.

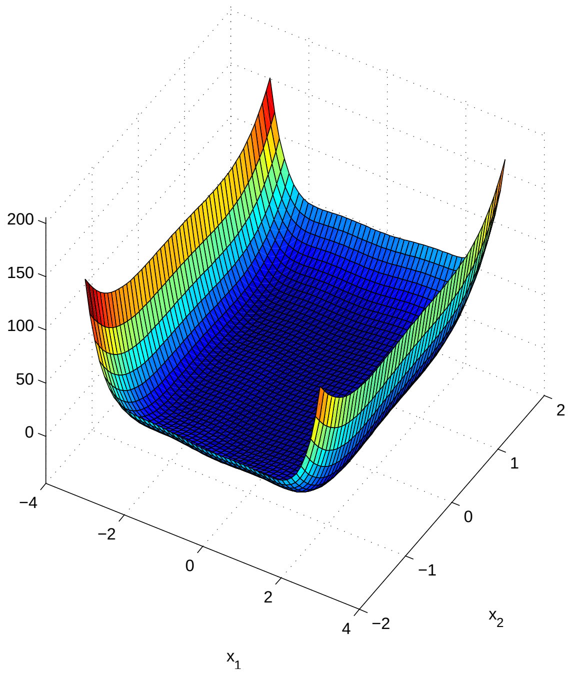

Figure 3.3: The six hump back camel optimisation test function

Shubert function

The Shubert function is two dimensional and has many local minima and 18 global minima.

$$f = \sum_{i=1}^{5} i \cos\left[(i+1)x_1 + i\right] \sum_{i=1}^{5} i \cos([(i+1)x_2 + i]) \tag{3.4.3}$$

with $x_1 \in [-10, 10], x_2 \in [-10, 10]$.

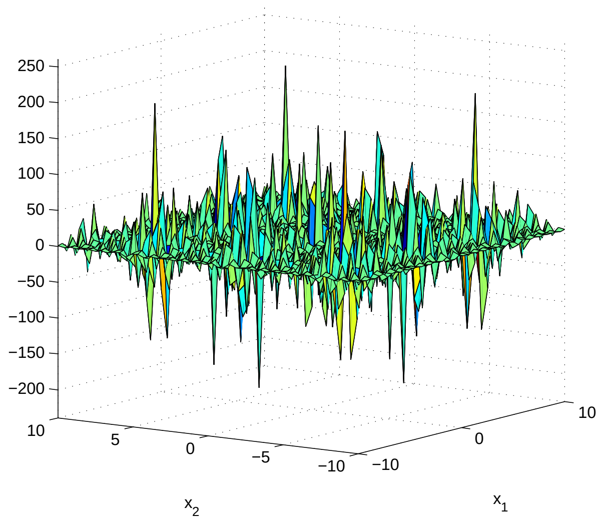

Figure 3.4: The Shubert optimisation test function

Goldstein-Price function

The Goldstein-Price function [16] is a well-known two dimensional global optimisation function. It contains three local minima and a single global minimum of $f = 3.0$ at [0,-1]. It is calculated as follows:

$$f = ab, \tag{3.4.4}$$

where

$$a = (1 + (x_1 + x_2 + 1)^2(19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1x_2 + 3x_2^2)) \tag{3.4.5}$$

$$b = (30 + (2x_1 - 3x_2)^2(18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1x_2 + 27x_2^2)) \tag{3.4.6}$$

and $x_1 \in [-2, 2]$,$x_2 \in [-2, 2]$.

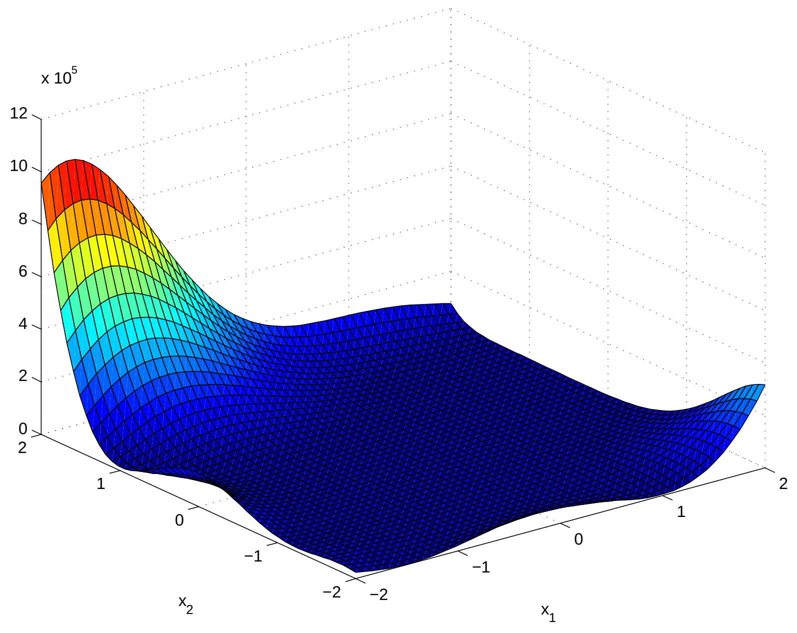

Figure 3.5: The Goldstein-Price optimisation test function

Shekel functions

The Shekel 5, Shekel 7 and Shekel 10 functions have 4, 6 and 9 local minima respectively and a single global optimum. All three functions are 4 dimensional and are calculated by

$$f = -\sum_{i=1}^{m} \frac{1}{(\mathbf{x} - \mathbf{A}_i)(\mathbf{x} - \mathbf{A}_i)^T + \mathbf{c}_i}, \tag{3.4.7}$$

where

$$\mathbf{A} = \begin{bmatrix} 4 & 4 & 4 & 4 \\ 1 & 1 & 1 & 1 \\ 8 & 8 & 8 & 8 \\ 6 & 6 & 6 & 6 \\ 3 & 7 & 3 & 7 \\ 2 & 9 & 2 & 9 \\ 5 & 5 & 3 & 3 \\ 8 & 1 & 8 & 1 \\ 6 & 2 & 6 & 2 \\ 7 & 3.6 & 7 & 3.6 \end{bmatrix}, \mathbf{c} = \begin{bmatrix} 0.1 \\ 0.2 \\ 0.2 \\ 0.4 \\ 0.4 \\ 0.6 \\ 0.3 \\ 0.7 \\ 0.5 \\ 0.5 \end{bmatrix}, \tag{3.4.8}$$

and $\mathbf{A}_i$, $\mathbf{c}_i$ return the i^{th} row of $\mathbf{A}$ and $\mathbf{c}$, respectively. The Shekel functions use their corresponding number as the value of m. The domain for all the Shekel functions is $\mathbf{x}_i \in [0, 10]$.

Hartman functions

The Hartman 3 and Hartman 6 functions are 3 and 6 dimensional global optimisation functions, respectively, both with bounds $\mathbf{x}_i \in [0, 1]$. They each have a single global optimum multiple local minima.

Rosenbrock's function

Rosenbrock's function is unimodal, but it useful to consider in global optimisation as the function is a deep valley with large gradient near the boundaries but very small gradients on the valley floor. The function can be defined in an Euclidean space $\Re^n$ as follows:

$$f = \sum_{i=1}^{n-1} (1 - x_i)^2 + 100(x_{i+1} - x_i)^2 \tag{3.4.9}$$

with bounds of $\mathbf{x}_i \in [-5.12, 5.12]$. Previous work has shown that mission design problems, particularly multiple gravity assist trajectories, contain valleys of solutions and therefore any appropriate global optimisation function should be able to optimise Rosenbrock's function. Increasing dimensionalities of 5, 10 and 15 were added to the test set.

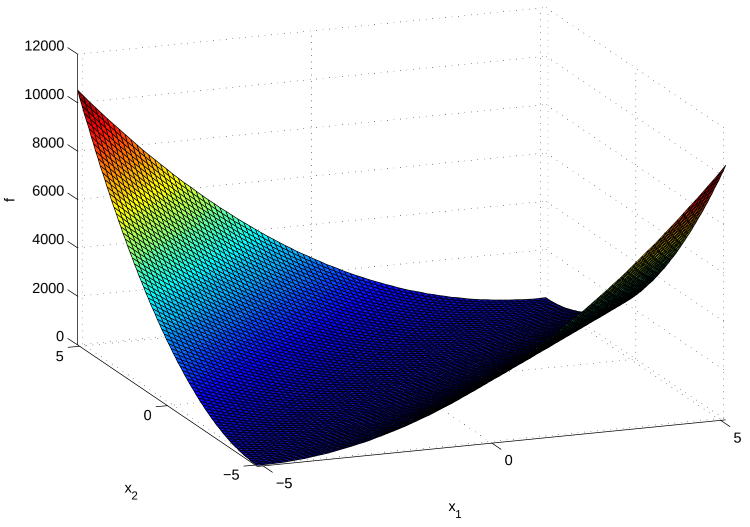

Figure 3.6: Rosenbrock's optimisation test function

Rastrigin's function

Rastrigin's function is a highly multimodal transdimensional function, but has only a single global optimum. However, the local minima are regularly spaced, and therefore could give unfair advantages to some algorithms. The original global optimum is at the origin but for the purposes of these experiments has been moved to $x = [1, 1, 1, \ldots]$. The reason for this is that some algorithms initialise by sampling the centre of the search space (such as DIRECT), and therefore would converge immediately and not give a true indication as to the algorithm's performance.

$$f = 10n + \sum_{i=1}^{n} (x_i - 1)^2 - 10 \cos 2\pi (x_i - 1). \tag{3.4.10}$$

The bounds for this function are the same as the Rosenbrock function: $x_i \in [-5.12, 5.12]$. Increasing dimensionalities of 5 and 10 were added to the test set.

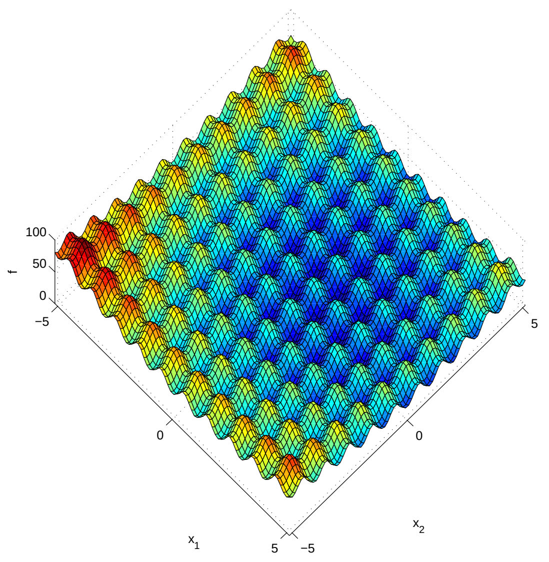

Figure 3.7: Rastrigin's optimisation test function

3.4.2 Test conditions

The algorithms were allocated a maximum of $4000n^2$ function evaluations for each test function, where n was the dimensionality of the problem. Each algorithm was run either until the located solution was within a threshold of 1×10^{-5} of the known global minimum or the allowed number of function evaluations was exceeded. For non-deterministic algorithms each test was performed 20 times in order to examine the range of solutions it provided. In all, the performance on 14 different test functions was evaluated for each algorithm. Due to differences in implementation languages, PGSLwas not applied to Shekel or Hartman functions.

3.4.3 Test results

The performance of the global optimisation algorithms on the benchmarking function is shown in tables 3.1 through to 3.14. Each table displays the minimum, maximum and median for both function evaluations required to converge and the distance from the global optimum. These measures are useful as they provide insight into how the algorithm performs both when it converges and when it does not.

For clarity, a check mark replaces the distance from the global optimum if that distance is within the given tolerance of 1×10^{-5}. This allows easy visual identification of which problems a given algorithm fails on.

A dash indicates that the algorithm was not applied to this test function because of unavailability of that test function in an appropriate language.

Table 3.1: The benchmark results for the Branin test function

	Evaluations			Distance from global optimum		
Name	Min	Max	Median	Min	Max	Median
MCS	85	85	85	✓	✓	✓
DIRECT	195	195	195	✓	✓	✓
DE	1000	2940	1880	✓	✓	✓
PSO	560	1120	700	✓	✓	✓
MPSO	660	4840	1300	✓	✓	✓
CE	1040	1520	1200	✓	✓	✓
GA	1240	16020	16020	✓	4.0008e-005	1.1289e-005
PGSL	880	4234	1767	✓	✓	✓

Table 3.2: The benchmark results for the Six-hump Camel test function

	Evaluations			Distance from global optimum		
Name	Min	Max	Median	Min	Max	Median
MCS	101	101	101	✓	✓	✓
DIRECT	177	177	177	✓	✓	✓
DE	840	2020	1590	✓	✓	✓
PSO	100	1280	660	✓	✓	✓
MPSO	540	2100	1260	✓	✓	✓
CE	880	2400	1280	✓	✓	✓
GA	1520	14080	5100	✓	✓	✓
PGSL	56	1096	773	✓	✓	✓

Table 3.3: The benchmark results for the Goldstein-Price test function

	Evaluations			Distance from global optimum		
Name	Min	Max	Median	Min	Max	Median
MCS	203	203	203	✓	✓	✓
DIRECT	305	305	305	✓	✓	✓
DE	1140	2000	1660	✓	✓	✓
PSO	560	1360	1020	✓	✓	✓
MPSO	1120	2980	1720	✓	✓	✓
CE	880	16000	1320	✓	0.4227	✓
GA	3720	16020	16020	✓	0.00076014	4.2653e-005
PGSL	898	1806	1363	✓	✓	✓

Table 3.4: The benchmark results for the Shubert test function

	Evaluations			Distance from global optimum		
Name	Min	Max	Median	Min	Max	Median
MCS	299	299	299	✓	✓	✓
DIRECT	16041	16041	16041	4.0664e-005	4.0664e-005	4.0664e-005
DE	3540	9100	5220	✓	✓	✓
PSO	1080	2260	1770	✓	✓	✓
MPSO	3400	16000	10030	✓	0.22398	✓
CE	3360	16000	4600	✓	0.18454	✓
GA	16020	16020	16020	1.3522e-005	0.062937	0.003256
PGSL	1590	13712	2748	✓	✓	✓

Table 3.5: The benchmark results for the Hartman 3 test function

	Evaluations			Distance from global optimum		
Name	Min	Max	Median	Min	Max	Median
MCS	196	196	196	✓	✓	✓
DIRECT	51199	51199	51199	✓	✓	✓
DE	1440	2220	1620	✓	✓	✓
PSO	840	1530	1215	✓	✓	✓
MPSO	1530	3540	2595	✓	✓	✓
CE	1200	1560	1440	✓	✓	✓
GA	1920	36030	7620	✓	1.1127e-005	✓
PGSL	-	-	-	-	-	-

Table 3.6: The benchmark results for the Shekel 10 test function

	Evaluations			Distance from global optimum		
Name	Min	Max	Median	Min	Max	Median
MCS	190	190	190	✓	✓	✓
DIRECT	4955	4955	4955	✓	✓	✓
DE	7080	13200	9060	✓	✓	✓
PSO	2520	64000	6820	✓	8.1147	✓
MPSO	8720	19720	14060	✓	✓	✓
CE	3200	64000	3360	✓	6.6332	✓
GA	51480	64040	64040	✓	7.6653	0.0010797
PGSL	-	-	-	-	-	-

Table 3.7: The benchmark results for the Shekel 5 test function

	Evaluations			Distance from global optimum		
Name	Min	Max	Median	Min	Max	Median
MCS	164	164	164	✓	✓	✓
DIRECT	317	317	317	✓	✓	✓
DE	6440	13960	9760	✓	✓	✓
PSO	2880	64000	64000	✓	7.5227	5.0752
MPSO	7840	19880	14400	✓	✓	✓
CE	3040	64000	3680	✓	7.5227	✓
GA	64040	64040	64040	7.6411e-005	7.5227	5.098
PGSL	-	-	-	-	-	-

Table 3.8: The benchmark results for the Shekel 7 test function

	Evaluations			Distance from global optimum		
Name	Min	Max	Median	Min	Max	Median
MCS	201	201	201	✓	✓	✓
DIRECT	4821	4821	4821	✓	✓	✓
DE	6160	13960	9220	✓	✓	✓
PSO	2520	64000	64000	✓	7.651	5.2741
MPSO	10280	19880	13860	✓	✓	✓
CE	3200	64000	3360	✓	6.6786	✓
GA	64040	64040	64040	3.8291e-005	7.6371	5.997
PGSL	-	-	-	-	-	-

Table 3.9: The benchmark results for the Hartman 6 test function

	Evaluations			Distance from global optimum		
Name	Min	Max	Median	Min	Max	Median
MCS	216	216	216	✓	✓	✓
DIRECT	11235	11235	11235	0.13455	0.13455	0.13455
DE	9840	45300	15960	✓	✓	✓
PSO	2880	144000	3690	✓	0.11921	✓
MPSO	23520	144000	46500	✓	0.0017043	✓
CE	3840	144000	4200	✓	0.11921	✓
GA	14280	144060	85410	✓	0.11921	0.059608
PGSL	-	-	-	-	-	-

Table 3.10: The benchmark results for the Rosenbrock 5 test function

	Evaluations			Distance from global optimum		
Name	Min	Max	Median	Min	Max	Median
MCS	135	135	135	✓	✓	✓
DIRECT	8345	8345	8345	✓	✓	✓
DE	55200	68400	62850	✓	✓	✓
PSO	3550	5900	4525	✓	✓	✓
MPSO	12850	25000	19825	✓	✓	✓
CE	100000	100000	100000	0.023658	10.6953	2.7557
GA	100050	100050	100050	0.00054007	14.67	3.0413
PGSL	4184	14906	8546	✓	✓	✓

Table 3.11: The benchmark results for the Rosenbrock 10 test function

	Evaluations			Distance from global optimum		
Name	Min	Max	Median	Min	Max	Median
MCS	310	310	310	✓	✓	✓
DIRECT	45763	45763	45763	✓	✓	✓
DE	400000	400000	400000	0.47992	1.5287	0.73752
PSO	17300	26700	21000	✓	✓	✓
MPSO	400000	400000	400000	1.5114e-005	0.056707	0.0066537
CE	400000	400000	400000	0.56349	23.1648	9.6636
GA	400100	400100	400100	0.65791	33.925	7.6539
PGSL	400000	400000	400000	0.0056635	0.8467	0.18643

Table 3.12: The benchmark results for the Rosenbrock 15 test function

	Evaluations			Distance from global optimum		
Name	Min	Max	Median	Min	Max	Median
MCS	563	563	563	✓	✓	✓
DIRECT	216093	216093	216093	✓	✓	✓
DE	900000	900000	900000	2.6686	10.9423	5.4643
PSO	900000	900000	900000	0.011476	6.0989	0.64084
MPSO	900000	900000	900000	0.00032344	15.0716	0.48488
CE	900000	900000	900000	0.37928	52.1148	13.0352
GA	900150	900150	900150	0.2946	29.1816	5.1513
PGSL	900000	900000	900000	0.14192	5.0832	2.8948

Table 3.13: The benchmark results for the Rastigrin 5 test function

	Evaluations			Distance from global optimum		
Name	Min	Max	Median	Min	Max	Median
MCS	405	405	405	✓	✓	✓
DIRECT	100159	100159	100159	1.9899	1.9899	1.9899
DE	10750	15350	12450	✓	✓	✓
PSO	6450	100000	100000	✓	6.9647	1.9899
MPSO	38300	100000	100000	✓	0.99496	0.99496
CE	7600	100000	8900	✓	0.99496	✓
GA	100050	100050	100050	0.0010724	0.070909	0.005654
PGSL	100000	100000	100000	0.99495	2.9849	0.99495

Table 3.14: The benchmark results for the Rastigrin 10 test function

	Evaluations			Distance from global optimum		
Name	Min	Max	Median	Min	Max	Median
MCS	217	217	217	✓	✓	✓
DIRECT	406873	406873	406873	8.0029	8.0029	8.0029
DE	107100	129600	117000	✓	✓	✓
PSO	400000	400000	400000	2.9849	25.8689	9.9496
MPSO	400000	400000	400000	0.99611	6.0754	2.9868
CE	19600	400000	26000	✓	0.99496	✓
GA	400100	400100	400100	0.00046412	0.0088303	0.0022301
PGSL	400000	400000	400000	1.9899	5.9697	4.9748

3.4.4 Summary

It is clear from the obtained results that MCS showed by far the best performance on the benchmark tests, even on the highly dimensional, highly multi-modal Rastrigin function. Of the stochastic algorithms, Differential Evolution was the most robust.

Although MPSO consistently outperformed PSO over most of the test functions, PSO was most effective on the Rosenbrock 10 function. However, this is to be expected as this function is unimodal and PSO is primarily a local optimiser - the added global exploration capabilities of MPSO were in this case unnecessary.

From these results, a ranking of these algorithms over a broad number of trials can be created. Firstly, they can be characterised in terms of the probability of converging successfully on a given problem over all trials. Where algorithms have not converged, the median error of such trials gives an indication of how close to convergence the algorithm was, or whether it had converged to a local minimum. Therefore, as a secondary value to compare algorithms with a similar convergence rate the mean value of these errors is presented.

Table 3.15 shows the summary table and the ranking of the all the considered algorithms. MCS ranks first, as expected, but interestingly the stochastic Differential Evolution and Multiple Particle Swarm Optimisation proved to be more effective than the other deterministic algorithm, DIRECT. In terms of non-convergence errors, CE produced the largest - this demonstrates that it has the largest tendency to converge to local minima, as had been observed throughout the experiments. PSO produced the second largest non-convergence error, as it too is known to get stuck in local minima.

All the algorithms proved to be more effective than the standard Genetic

Table 3.15: A summary of the performance of the global optimisation algorithms on the benchmark tests

Algorithm	% Converged	Mean of median non-convergence errors
MCS	100	0
DE	85.7	0.4429
MPSO	71.4	0.3197
DIRECT	71.4	0.7234
CE	68.6	3.4869
PSO	65.0	2.5347
PGSL	55.6	1.0057
GA	21.8	1.9921

Algorithm used for comparison, which only managed to converge in 21.8% of trials in the allowed number of function evaluations. However, the GA did outperform MPSO on the Rastrigin 10 function, which was the most difficult benchmark: it is hypothesised that this is due to the regular minima spacing that would make the crossover operation unusually effective - any two individuals that lay at the bottom of different minima would always produce a child also at the bottom of a minimum.

Therefore, the conclusion of this benchmarking is that MCS, DE and MPSO are the most promising global optimisation algorithms.

Shortcomings of Rastrigin function

Although the Rastrigin function is popular for testing global optimisation functions, some major shortcomings have been identified during the course of experimentation. The main one is that the parameters are completely independent, a weakness not shared by the Rosenbrock function. Therefore, performing a 1D global optimisation over one parameter, regardless of the values of any other parameters, *will* yield the globally optimum value for that parameter. Consequently, a conjugate gradient method would give very good results with few function evaluations.

In real mission design problems the parameters are certainly *not* independent, and therefore a good performance of an algorithm on the Rastrigin function will not necessarily correlate with good performance on mission design problems.

Properties of MCS

Experimentally, MCS proved to be by far the most efficient algorithm over all the test functions. However, it will be shown that subsequent tests on real mission design problems fail to replicate this achievement. It is suspected that MCS has been specifically tuned to function very well on standard test functions, both in terms of initialisation parameters and branching heuristics, but as a result is much more prone to getting stuck in local minima on real problems.

The following function was constructed as a trivial modification of the Rastrigin function with exactly the same properties

$$f = 10n + \sum_{i=1}^{n} (x_i - 3)^2 - 10 \cos 10\pi (x_i - 3). \tag{3.4.11}$$

The oscillation of the function has been increased by a factor of 5, and the global optimum has been moved to $\mathbf{x} = [3, 3, 3 \ldots, 3]$.

MCS now fails to converge within the 100000 function evaluations even in the 5 dimensional case and terminates with a best solution of 0.31994. DE, however, can still converge successfully in this case.

PGSL performed the second most poorly on the test functions, and thus will be disregarded for application to real mission design problems. Although performing the least effectively, the standard genetic algorithm will be retained for comparison due to it's popularity in the field of mission design.

3.5 Global Optimisation of Thrust Optimal Interplanetary Transfers

This section examines the global optimisation of a thrust optimal Earth-Mars transfer including a braking manoeuvre at Mars. The control is bi-impulsive. The decision variables are the launch date, t_0 (in MJD2000) and t, the mission time, in days. This is the same problem as described in [47], although with different bounding constraints. The position of the spacecraft is denoted $\mathbf{x}$, and the velocity of the spacecraft $\dot{\mathbf{x}}$.

The objective function f to be investigated is the total $\triangle\mathbf{V}$ required in order to effect the interplanetary transfer:

$$f = |\dot{\mathbf{x}}_i| + |\dot{\mathbf{x}}_f|, \tag{3.5.1}$$

where $\dot{\mathbf{x}}_i$ is the initial hyperbolic excess velocity (relative to Earth) and $\dot{\mathbf{x}}_f$ is the velocity relative to Mars on entry to Mars' sphere of influence. $\dot{\mathbf{x}}_i$ and $\dot{\mathbf{x}}_f$ are determined for a given t_0 and t by solving the associated Lambert problem [2, 17, 10]. The bounds on the decision variables were chosen such that several synodic periods would be present. Multiple revolution solutions to the Lambert problem were not considered. The bounds were

- $t_0 \in [800, 3800]$ MJD2000
- $t \in [100, 400]$ days.

3.5.1 Results

For the population based algorithms (DE, PSO and MPSO) 50 individuals were allocated rather than the 20 that would be used according to the standard scheme. This was because the local minima are quite isolated within the search space of this problem, and a small population increases the probability of convergence to a local minima.

Due to the low dimensionality of this problem the global optimum can be reliably identified as 5667.1964m/s.

Table 3.16 show the results obtained after applying all the global optimisation algorithms to this problem. A checkmark is again used to indicate convergence to the global optimum within a tolerance of 1×10^{-5}m/s. Out of

Table 3.16: Algorithm performance on bi-impulsive Earth-Mars transfer

	Evaluations			Best solution (km/s)		
Name	Min	Max	Median	Min	Max	Median
MCS	126	126	126	✓	✓	✓
DIRECT	12009	12009	12009	5667.1997	5667.1997	5667.1997
DE	3550	9050	6775	✓	✓	✓
PSO	2800	12000	4025	✓	5669.5491	✓
MPSO	3200	12000	6150	✓	5667.2032	✓
CE	12000	12000	12000	5669.5491	6143.3754	5753.6501
GA	12020	12020	12020	5667.1998	5670.1203	5668.8033

all the global optimisation functions, only DIRECT and CE did not manage to locate the global optimum to 4.d.p. at least once in the allowed number of function evaluations. Out of the stochastic functions, only DE located the global minimum to the required tolerance in every trial.

Therefore, for bi-impulsive interplanetary transfers, the MCS algorithm is recommended. If a stochastic algorithm were to be applied, though, DE would be the best choice.

In the final application deliverable DE will be provided as an optimiser for this problem rather than MCS. Although MCS has proved itself to be more effective on this problem, it is a complex algorithm that is unavailable in C++ and would take an impractical amount of time to reliably convert under given timescales. DE, however, is a much simpler algorithm and conversions to C++ are already available.

In practical terms, the number of function evaluations required by DE would still correspond to negligible run time on a standard PC and so implementing the more effective MCS would not make a perceivable difference.

3.6 Global Optimisation of Multiple Gravity Assist Trajectories

The test case for Multiple Gravity Assist trajectories will be the Cassini-Huygens mission.

This section considers the Cassini-Huygens Earth-Saturn mission as a test case. The following mathematical models and assumptions were used:

- 2 body dynamics (only the gravitational attraction of the Sun is considered).
- Elliptic, non-coplanar orbits using analytical Ephemeris.
- Point-mass spacecraft assumption.
- Link-conic approximation for gravity assist.

Gobetz's method [15] was used to determine the appropriate thrust at the hyperbolic periapse to transfer between the desired incoming and outgoing hyperbolic asymptotes. Swingbys where the periapse radius is less than 110% of the planetary radius are considered to be infeasible.

The orbital injection will remain the same as the original Cassini-Huygens mission, with the following parameters:

- Eccentricity, $e = 0.98$
- Radius of periapse, $r_p = 1.0895 \times 10^5$km

Orbital injection is achieved with a single impulsive braking thrust at the periapse of the hyperbolic orbit, thus creating a highly eccentric elliptic orbit.

The desired angular momentum h of the orbit can be calculated simply as follows, by finding the semi-major axis a, and the semi-latus rectum l

$$a = \frac{r_p}{1-e} \tag{3.6.1}$$

$$l = \frac{a}{1-e^2} \tag{3.6.2}$$

$$h = \sqrt{lGM}, \tag{3.6.3}$$

where G is the gravitational constant and M is the mass of Saturn. The magnitude of the velocity at periapse, $\mathbf{v}_{pf}$, is then simply

$$|\mathbf{v}_{pf}| = \frac{h}{r_p}. \tag{3.6.4}$$

The magnitude of the velocity the probe will achieve at the periapse of the orbit, $|\mathbf{v}_{pi}|$, can be calculated simply from the transferral of potential energy to kinetic energy, so

$$|\mathbf{v}_{pi}| = \sqrt{|\mathbf{v}_o|^2 + \frac{2GM}{r_p}}, \tag{3.6.5}$$

and therefore the braking manoeuvre is of the magnitude

$$|\triangle\mathbf{V}| = |\mathbf{v}_{pi}| - |\mathbf{v}_{pf}| \tag{3.6.6}$$

The objective function was the total $\triangle\mathbf{V}$, including launch thrust, of the entire mission, so was of the form

$$f = \triangle\mathbf{V}_1 + \triangle\mathbf{V}_{GA1} + \triangle\mathbf{V}_{GA2} + \ldots + \triangle\mathbf{V}_{\text{insertion}}. \tag{3.6.7}$$

Although this objective function is not the most useful in practical mission design, as the C3 will be provided by a launcher, it is good for testing the optimisational capabilities of the global algorithms for producing optimal missions.

A more useful objective function for MGA mission design would be to optimise the probe thrust alone with some kind of penalty term added if the

C3 was over some acceptable limit. A further improvement on this would be the inclusion of launcher characteristics into the objective function, thus allowing the scientific payload itself to be maximised which is, in the majority of cases, the *true* objective function. When creating the final application for optimisation of multiple gravity assist missions all these objective functions will be included as options.

Direct comparison with results from literature is problematic, as the Ephemeris used, minimum swingby periapse and swingby model all significantly influence the global minimum of the search space. For example, reducing the minimum swingby radius to 100% of the planetary radius in the EVEJS case from 110% yields over a 200m/s difference in the global minimum. Consequently, the results from this report will be directly used to assess the final optimisation product in terms of accuracy and efficiency.

3.6.1 Algorithm Initialisation and Test Conditions

Each algorithm was initialised as follows, in order to maximise the repeatability of these experiments:

MCS Default initialisation

DIRECT Default initialisation

DE Population = $10n$, Crossover prob = 0.5, Scale = 0.8

PSO Population = $10n$, $\omega = 0.65$, $\eta_1 = 2.0$, $\eta_2 = 2.0$

MPSO Population = $10n$, Clusters = $3n$, $\omega = 0.65$, $\eta_1 = 2.0$, $\eta_2 = 2.0$

CE Population = $40n$, $\alpha = 0.7$, $\beta = 0.8$, $q = 5$, μ placed in centre of search space, σ equal to half search space size.

GA Population = $10n$, Crossover prob = 0.5, Perturbance prob = 0.1

A limit of $4000n^2$ function evaluations was applied to each algorithm, where n is the dimensionality of the problem.

Multiple Gravity Assist problems are significantly more complex than interplanetary transfers, and as a consequence in the higher dimensional cases it is difficult to know the exact global optimum for a given set of mathematical models. However, preliminary trials on the EJS and EMJS case have reliably yielded a single best solution which will be assumed to be the global optimum. Although it would be infeasible to prove that these are the true global minimum there is a very high probability that this is the case. The reason for this assumption is that the speed of convergence can be assessed for each algorithm on the simpler problems.

On the EVEJS and EVVEJS, though, each algorithm will be run for a terminal number of function evaluations and the algorithm that consistently produces the best results will be considered the best algorithm, although in this case it is unknown whether the true global optimum has been located.

3.6.2 Earth-Jupiter-Saturn transfer

The simplest case, and the one previously analysed in Work Package 2, is the Earth-Jupiter-Saturn transfer. The search space for this problem is 3 dimensional and the form of the decision variable vector $\mathbf{x}$ is

$$\mathbf{x} = [t_0, t_1, t_2], \tag{3.6.8}$$

where t_0 is the launch date, t_1 is the Earth-Jupiter transfer time and t_2 is the Jupiter-Saturn transfer time. The bounds on the decision variables were:

- $t_0 \in [-1278, 547]$ MJD2000

- $t_1 \in [99.71994.8]$ days
- $t_2 \in [366.07320.9]$ days

The best solution found in preliminary trials by DE, PSO and MPSO showed reliable convergence to a minimum of 9351.7902m/s, and this was taken as being the global optimum.

The stopping criteria for the algorithms was defined as reaching within a tolerance 1m/s of the global minimum. Table 3.17 shows the obtained results by applying the selected global algorithms to this problem. Although performing fantastically well on benchmark problems, here MCS locates the worst solution of all the algorithms, at 9419.1137m/s. The best performing algorithm here is DIRECT. Out of the stochastic algorithms, DE was ranked first although it did not located the global optimum most robustly, but not in all trials. PSO, MPSO and CE also located the global optimum at least once.

Figure 3.8 shows the trajectory corresponding to the globally optimum EJS transfer.

Table 3.17: Algorithm performance on EJS transfer

	Evaluations			Best solution (km/s)		
Name	Min	Max	Median	Min	Max	Median
MCS	18019	18019	18019	9419.1137	9419.1137	9419.1137
DIRECT	8101	8101	8101	✓	✓	✓
DE	4350	18000	7230	✓	9417.8245	✓
PSO	1440	18000	18000	✓	9573.9828	9417.8245
MPSO	1710	18000	18000	✓	9426.1464	9359.0939
CE	3600	18000	18000	✓	9603.0661	9422.3257
GA	18030	18030	18030	9352.8717	9716.5473	9431.0415

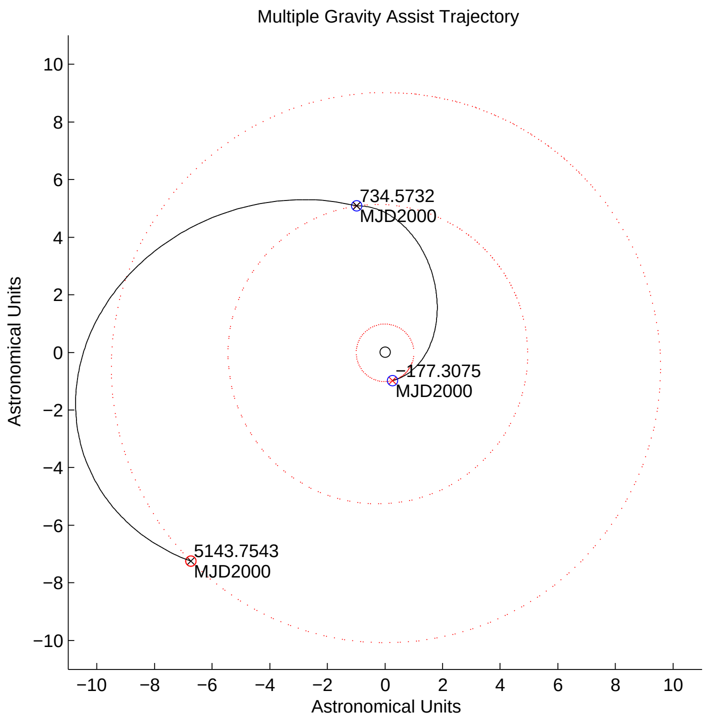

Figure 3.8: The globally optimum trajectory for an EJS transfer

Decision Variables	
t_0(MJD2000)	-177.3075
t_1(days)	911.8806
t_2(days)	4409.1811
Objectives	
Mission Time (days)	5321.0617
$\triangle\mathbf{V}_E$(m/s)	8917.8984
$\triangle\mathbf{V}_S$(m/s)	433.8919
$\triangle\mathbf{V}_{GA}$(m/s)	2.2216e-4

It can be seen that these values match closely to those found in Work Package 2 for an unpowered EJS transfer, although the match is not exact as the Ephemeris routines have been updated to correct a minor error in planetary velocity calculation. Approximately 15% of the probe weight here would be fuel assuming an I_{sp} of 300s.

3.6.3 Earth-Mars-Jupiter-Saturn transfer

The next trajectory to consider is that of the Earth-Mars-Jupiter-Saturn. The search space for this problem is 4 dimensional and the form of the decision variable vector $\mathbf{x}$ is

$$\mathbf{x} = [t_0, t_1, t_2, t_3], \tag{3.6.9}$$

where t_0 is the launch date, t_1 is the Earth-Mars transfer time, t_2 is the Mars-Jupiter transfer time and t_3 is the Jupiter-Saturn transfer time. The bounds on the decision variables were:

- $t_0 \in [-1278, 547]$ MJD2000
- $t_1 \in [25.91035.5]$ days
- $t_2 \in [112.62252.7]$ days
- $t_3 \in [366.07320.9]$ days

Table 3.18 shows the obtained results by applying the selected global algorithms to this problem. Again, MCS performed poorly, highlighting the brittle nature of the algorithm. The best performing algorithm was MPSO, which converged on over 50% of trials, with PSO just behind. DE performed the third best and CE the fourth. Overall, the best solution found was 9185.9709m/s.

Figure 3.8 shows the trajectory corresponding to the globally optimum EMJS transfer.

Table 3.18: Algorithm performance on EMJS transfer

	Evaluations			Best solution (km/s)		
Name	Min	Max	Median	Min	Max	Median
MCS	32019	32019	32019	9756.1777	9756.1777	9756.1777
DIRECT	32027	32027	32027	11749.4932	11749.4932	11749.4932
DE	22840	32000	32000	✓	9447.7202	9283.1935
PSO	2640	32000	32000	✓	10437.2662	9236.4378
MPSO	10360	32000	17340	✓	9283.104	✓
CE	32000	32000	32000	9189.0797	9690.2242	9232.9727
GA	32040	32040	32040	9197.0195	12188.6793	10166.5942

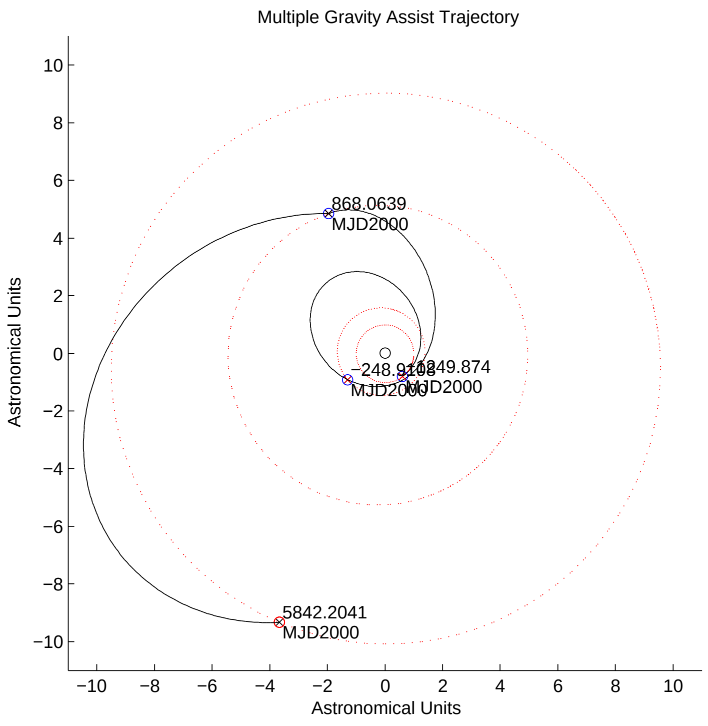

Figure 3.9: The globally optimum trajectory for an EMJS transfer

Decision Variables	
t_0(MJD2000)	-1249.87396872206
t_1(days)	1000.96320973301
t_2(days)	1116.97463451417
t_3(days)	4974.14021342797
Objectives	
Mission Time (days)	7092.0781
$\triangle\mathbf{V}_E$(m/s)	7067.8258
$\triangle\mathbf{V}_M$(m/s)	1684.3675
$\triangle\mathbf{V}_J$(m/s)	0.0000
$\triangle\mathbf{V}_S$(m/s)	433.7777

The optimum EMJS mission requires significantly less C3 than the corresponding EJS transfer ($50\text{km}^2/\text{s}^2$ compared to $79\text{km}^2/\text{s}^2$), thus increasing the mass of probe which could be launched, but requires over 4 times as much $\triangle\mathbf{V}$ by the probe. The proportion of the probe that would be fuel would be approximately 50% (assuming an I_{sp} of 300s).

3.6.4 Earth-Venus-Earth-Jupiter-Saturn transfer

The third sequence of planets to be considered is Earth-Venus-Earth-Jupiter-Saturn. This results in a 5 dimensional search space. The bounds on the variables were:

- $t_0 \in [-1278, 547]$ MJD2000
- $t_1 \in [14.6584.3]$ days
- $t_2 \in [14.6584.3]$ days
- $t_3 \in [99.71994.9]$ days
- $t_4 \in [366.07320.9]$ days

Preliminary results had not shown repeated convergence to a given best minima, and therefore each algorithm was run to a terminal number of iterations and the final solutions recorded. Table 3.19 shows the results obtained: The best known optimum for this problem is 7548.3646m/s (from the preliminary trials), which was not improved upon in this experiment although MPSO reached within a tolerance of 0.3m/s of it, and PSO within 1.3m/s of it. There is a significant probability that this value is close the global optimum, although it is much less certain than in the EJS and EMJS cases.

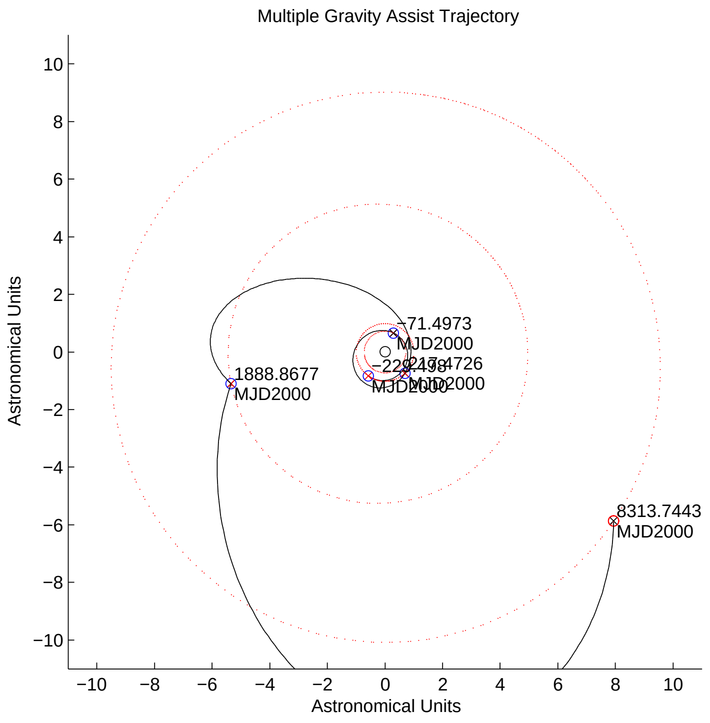

Figure 3.10: The best found trajectory for an EVEJS transfer

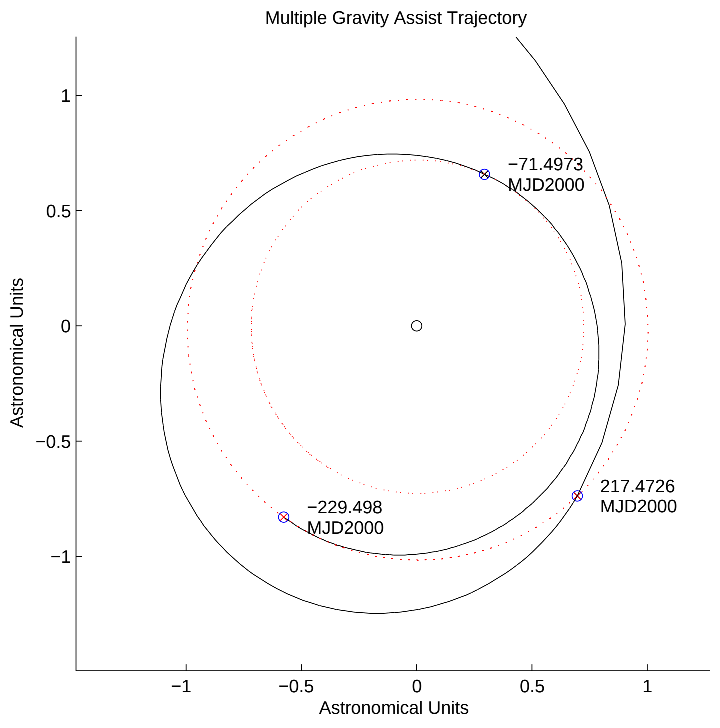

Figure 3.11: A close up on the EVE part of the EVEJS transfer

Table 3.19: Algorithm performance on EVEJS transfer

	Evaluations			Distance from global optimum(km/s)		
Name	Min	Max	Median	Min	Max	Median
MCS	50023	50023	50023	11938.1793	11938.1793	11938.1793
DIRECT	50017	50017	50017	9719.8932	9719.8932	9719.8932
DE	50000	50000	50000	7600.2166	8925.9367	7704.702
PSO	50000	50000	50000	7549.0635	16854.074	8456.0771
MPSO	50000	50000	50000	7548.5853	8229.8574	7713.8165
CE	50000	50000	50000	7716.8575	9762.8498	8642.0969
GA	50050	50050	50050	8012.8772	14114.8682	11602.9307

Decision Variables	
t_0(MJD2000)	-229.497976766197
t_1(days)	158.00066587151
t_2(days)	288.96992264231
t_3(days)	1671.39505180009
t_4(days)	6424.87661250939
Objectives	
Mission Time (days)	8543.2423
$\triangle\mathbf{V}_E$(m/s)	2960.676
$\triangle\mathbf{V}_V$(m/s)	0.3259
$\triangle\mathbf{V}_E$(m/s)	3976.4275
$\triangle\mathbf{V}_J$(m/s)	0.0180
$\triangle\mathbf{V}_S$(m/s)	611.1379

Again, the best solution with respect to overall $\triangle\mathbf{V}$ is longer in the EVEJS case than in the EMJS and EJS cases. However, the C3 has been decreased significantly to 8km^2/s^2 from 50km^2/s^2 in the EMJS case, so the spacecraft can have a much larger launch mass. Again, the required probe $\triangle\mathbf{V}$ has increased to about 4600m/s, and this means that approximately 80% of the spacecraft will consist of fuel (assuming an I$_{sp}$ of 300s). The trajectory is shown in Figures 3.10 and 3.11.

3.6.5 Earth-Venus-Venus-Earth-Jupiter-Saturn transfer

The final gravity assist sequence to be investigated will be Earth-Venus-Venus-Earth-Jupiter-Saturn: this is equivalent to the Cassini-Huygens mission without the Deep Space Manoeuvre between the two successive Venus swingbys. This problem yields a 6 dimensional search space. The bounds on the variables were:

- $t_0 \in [-1278, 547]$ MJD2000
- $t_1 \in [14.6584.3]$ days
- $t_2 \in [22.5898.7]$ days
- $t_3 \in [14.6584.3]$ days
- $t_4 \in [99.71994.8]$ days
- $t_5 \in [366.07320.9]$ days

Table 3.20: Algorithm performance on EVVEJS transfer

	Evaluations			Distance from global optimum(km/s)		
Name	Min	Max	Median	Min	Max	Median
MCS	72048	72048	72048	15771.2693	15771.2693	15771.2693
DIRECT	69831	69831	69831	12678.9257	12678.9257	12678.9257
DE	72000	72000	72000	6141.9986	13463.4896	8419.0533
PSO	72000	72000	72000	6971.814	32268.5934	17105.7534
MPSO	72000	72000	72000	7834.3541	14772.6228	11797.0875
CE	72000	72000	72000	17175.5585	25583.5021	20109.437
GA	72060	72060	72060	9183.1234	30484.8734	16105.2964

DE located the best minima at 6142.00m/s (see Figures 3.12 and 3.13 and also performed the consistently the best on the problem - PSO was ranked

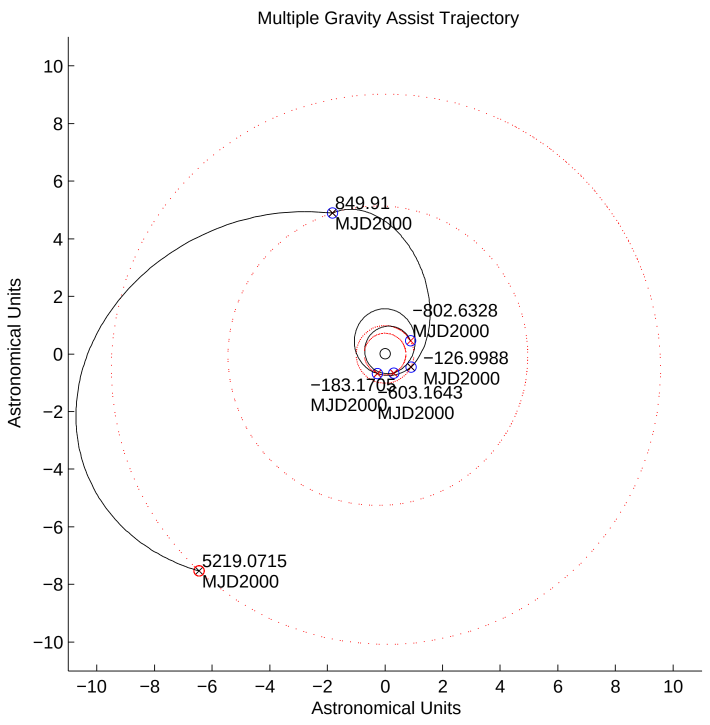

Figure 3.12: The best found trajectory for an EVVEJS transfer

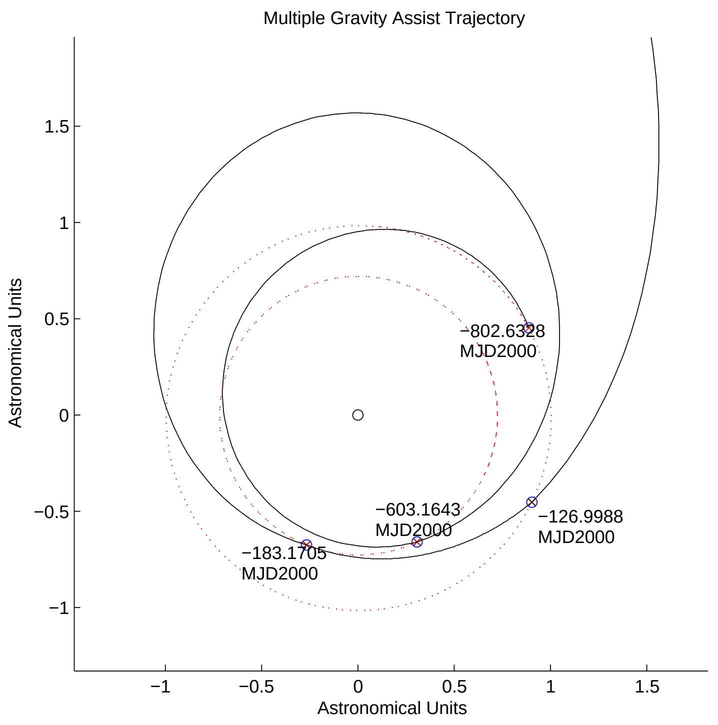

Figure 3.13: A close up on the EVVE section of the EVVEJS transfer

second in this case and MPSO third in terms of best solution located. In the preliminary trials, however, MPSO found a better solution of 5986.75m/s.

For this case further experimentation has shown that the global optimum in this problem is less than 5400m/s, and hence it is certain that none of the algorithms have converged correctly. Analysing the range of solutions over the trials shows that the algorithms did not produce the same solutions more than once, indicating that many more function evaluations would be required on such problems in order to obtain proper convergence.

However, these results still give an indication as to the speed of convergence of the algorithms on the problem, and this rank order compares well to the rank order of convergence on problems with known global optima.

3.6.6 Summary

For multiple gravity assist trajectories without a Deep Space Manoeuvre, the MPSO algorithm consistently found better solutions than the other algorithms over the 20 trials. However, DE consistently produced a significantly smaller range of results and can therefore be considered more robust.

3.7 Global Optimisation of Multiple Gravity Assists with Deep Space Manoeuvres

Deep Space Manoeuvres can be used to reduce thrust requirements in multiple gravity assist trajectories, as in the case of the Cassini-Huygens mission. The swingby before the DSM is forced to be unpowered, with the periapse radius and B-plane now decision variables. The orbit is then analytically propagated using a universal variables formulation [3] up until the DSM, at which point the Lambert problem point from the DSM point to the next gravity assist planet is solved. Consequently, using a single DSM introduces

3 additional decision variables:

- $1/r_p$ - the reciprocal of the periapse radius. r_p is measured in appropriate planetary radii. The reciprocal is used so that the decision variable can be easily bounded between 0 and 1, where 1 is a swingby with minimum acceptable periapse radius and 0 is a swingby at infinity. $1/r_p \in [0, 1]$.
- ϕ - The deflection from the planet is initially calculated in the plane formed by the incoming velocity and heliocentric position vector of the planet. ϕ then defines the rotation of the outgoing velocity using the incoming planet relative velocity as a rotational axis. $\phi \in [-\pi, \pi]$
- α defines the point in the trajectory where the DSM is applied. It is the proportion of the time until the next desired gravity assist. $\alpha \in [0.1, 0.9]$.

The global optimisation of two examples will be considered here - an EJS transfer with a Deep Space Manoeuvre between Jupiter and Saturn, and an EVVEJS transfer with a DSM between the two Venus swingbys (the Cassini-Huygens mission).

The number of function evaluations allowed for each algorithm was $4000n^2$, with n being the problem dimensionality.

3.7.1 EJS With Deep Space Manoeuvre

Consider first the addition of a DSM to the EJS transfer - adding a DSM may well increase the flexibility of the mission. The overall search space is now 6 dimensional.

The best solution obtained here by DE is identical to that found for the same transfer without a DSM - 9351.8m/s. Therefore, it seems that adding a

Table 3.21: Algorithm performance on EJS transfer with DSM

	Evaluations			Best solution (km/s)		
Name	Min	Max	Median	Min	Max	Median
MCS	144035	144035	144035	9566.0357	9566.0357	9566.0357
DIRECT	144007	144007	144007	9418.2578	9418.2578	9418.2578
DE	144000	144000	144000	9351.7902	10370.7557	9616.3622
PSO	144000	144000	144000	9417.8245	11243.5548	9630.8354
MPSO	144000	144000	144000	9352.095	9953.2427	9616.3622
CE	144000	144000	144000	10075.6434	11394.8907	10604.3133
GA	144000	144000	144000	9472.9115	11204.2255	10366.1252

DSM to this trajectory does not decrease the thrust required, and therefore the DSM can be considered redundant in this case. The redundancy of the DSM can also be observed in Figure 3.14, which can be compared exactly to the non-DSM trajectory in Figure 3.8.

Only DE managed to locate the minimum to 4 d.p. in one trial out of all the algorithms, although MPSO found it to within an acceptable tolerance of 0.5m/s. Observing the converged values for DE and MPSO over all the trials show that local minima exist near the global optimum with values f = 9616.36m/s, f = 9417.82m/s, and f = 9523.57m/s - these values were converged to in more than one trial and the probability of this occurring if these values were not local minima is negligible.

3.7.2 EVVEJS with Deep Space Manoeuvre

Adding a DSM to the EVVEJS trajectory yields a 9 dimensional search space. The bounds on t_0 to t_5 are the same as those for the EVVEJS transfer without DSM considered in 3.6.5. Unlike the EJS trajectory, adding a DSM to the EVVEJS transfer has obvious advantages, as the best solution has increased significantly from 5986m/s to 4589m/s (as found by DE), although MPSO too located a better trajectory than previously known. The statistics of the best solution (located by DE) were as follows:

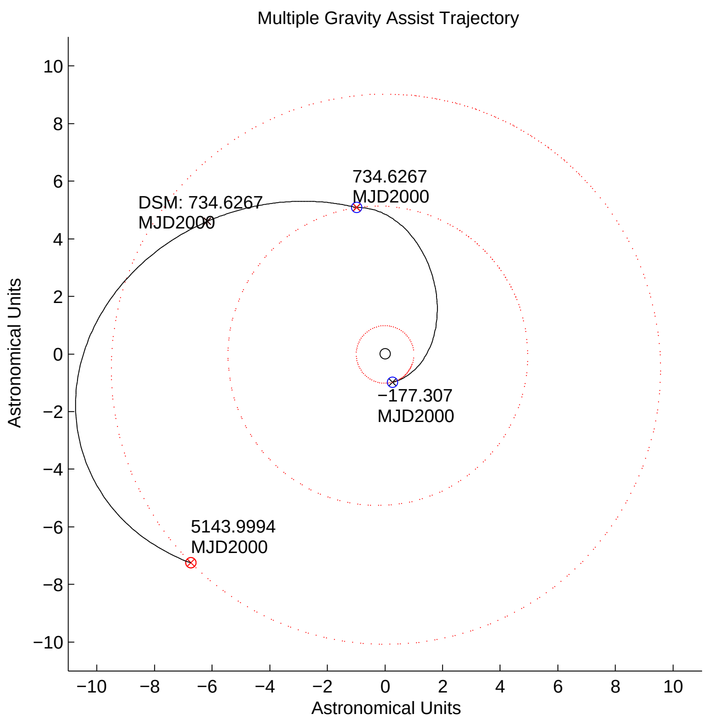

Figure 3.14: The optimal EJS transfer with a DSM

Table 3.22: Algorithm performance on EVVEJS transfer with DSM

	Evaluations			Best solution (km/s)		
Name	Min	Max	Median	Min	Max	Median
MCS	324000	324000	324000	9428.5978	9428.5978	9428.5978
DIRECT	324019	324019	324019	13923.3797	13923.3797	13923.3797
DE	324000	324000	324000	4589.0632	15474.594	6615.9064
PSO	324000	324000	324000	6516.9291	20456.933	11038.169
MPSO	324000	324000	324000	5217.6691	12860.2774	7489.6959
CE	324000	324000	324000	10270.8495	28923.1056	13983.1441
GA	324000	324000	324000	7330.348	21068.3104	13762.2777

Decision Variables	
t_0(MJD2000)	-796.4637
t_1(days)	188.1997
t_2(days)	428.5360
t_3(days)	52.9901
t_4(days)	1004.4382
t_5(days)	4508.6936
ϕ	-0.79218925
$1/r_p$	0.85811739
α	0.49116994
Objectives	
Mission Time (days)	6183.4147
$\triangle\mathbf{V}_E$(m/s)	3900.2989
DSM(m/s)	220.9456
$\triangle\mathbf{V}_V$(m/s)	0.23693m/s
$\triangle\mathbf{V}_E$(m/s)	0.076045m/s
$\triangle\mathbf{V}_J$(m/s)	0.42476m/s
$\triangle\mathbf{V}_S$(m/s)	467.0809m/s

This mission, shown in Figures 3.15 and 3.16, is significantly better than the actual Cassini-Huygens mission in terms of every objective apart from mission time - it takes over 3 times as long, although it is still shorter than the best EJS, EMJS and EVEJS trajectories found that did not use a DSM. However, all the swingbys are effectively unpowered, only a small DSM is required and the C3 is relatively low (at 15.21km^2/s^2). As less than 900m/s $\triangle\mathbf{V}$ is required by the probe, a much larger scientific payload could be achieved. In comparison, over half of the mass of the Cassini-Huygens craft was fuel.

3.7.3 Summary

The results show that Differential Evolution is the most effective algorithm for optimising multiple gravity assist transfers with a deep space manoeuvre, with Multiple Particle Swarm Optimisation being ranked second. However, even with a very large amount of function evaluations (324000) neither were able to consistently find the global optimum.

Following the results from the non-DSM transfers investigated, it seems a valid assumption that the algorithms which locate the best solutions are also significantly more likely to converge to the global minimum in the more complex transfer problems. For example, in the EJS transfers with DSM, only DE and MPSO converged within an acceptable tolerance to the (assumed) global minimum, and these two algorithms also found the best solutions in the more complex EVVEJS case, even though they failed to converge. Therefore, it is likely that if the number of allowed function evaluations was increased the same two algorithms would be most likely to converge.

3.8 Conclusions

This work package has investigated the performance of global optimisation algorithms both on a representative set of well known benchmark functions for simple comparison and also on two relevant mission design problems: interplanetary transfers and multiple gravity assist trajectories (both with and without deep space manoeuvres). The genetic algorithm, included for it's popularity in mission design, performed poorly in comparison to modern robust estimation techniques such as Differential Evolution and Particle Swarm Optimisation. Similarly, the deterministic algorithms investigated,

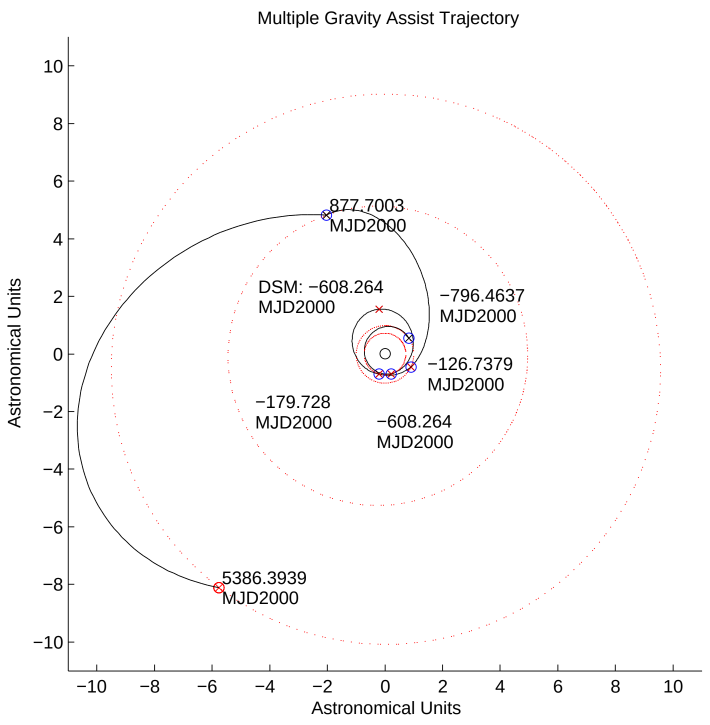

Figure 3.15: The best found trajectory for an EVVEJS transfer with DSM

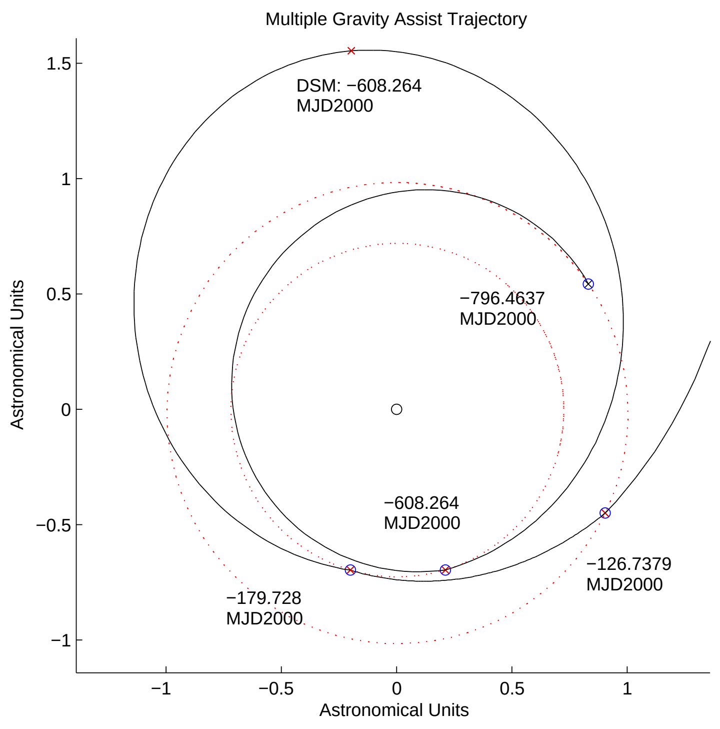

Figure 3.16: A close up on the EVVE section of the EVVEJS transfer with DSM

Table 3.23: Summary of investigation

Mission Design Problem	Primary Selection	Secondary Selection
Interplanetary Transfer	MCS	DE
MGA transfer (without DSM)	MPSO	DE
MGA transfer (with DSM)	DE	MPSO

DIRECT and MCS, did not perform well on the more complex MGA trajectory design problems, even though theoretically they are much more robust than stochastic methods.

Table 3.23 summarises the findings and selection of algorithm for each mission design problem: Differential Evolution was ranked either first or second for all three considered problems, and hence will be given the highest priority for inclusion in the final application deliverable.

From the Multiple Gravity Assist results, it is clear that even modern global optimisation algorithms that perform extremely well on benchmarks find it difficult to optimise such trajectories efficiently and consistently. Since Multiple Gravity Assist trajectories are of great interest in trajectory design, both due to their ability to reduce fuel expenditure and also increase scientific yield from analysis of the swingby planets, it would be of great interest to develop a novel algorithm that could effectively optimise this specific problem. From the analysis in this report and from Work Package 2, it has been concluded that the vast majority of the MGA search space consists of completely infeasible solutions i.e. those that have impractical C3 or require enormous thrusts during gravity assists. Therefore, it seems that a method that could efficiently prune the MGA search space through the use of domain knowledge, such that only good solutions were left, would be very useful. Conventional global optimisation methods could then be applied much more effectively to the pruned search space as the basin of attraction of the global minimum would be much larger, proportional to the size of the search space. This approach is in keeping with the methodology of Törn [44], who recommends minimising the size of the search space wherever possible. Usually, this would mean only considering small ranges of launch dates and transfer times, but with effective pruning large launch windows could be efficiently considered.

Work Package 4, which concerns the development of novel global optimisation tools, will concentrate on the possibility of this type of approach with a view to creating a method that can quickly and efficiently prune the MGA trajectory space. Further work could consider expanding the scope of this technique to allow deep space manoeuvres and low thrust arcs.

Chapter 4

Development of Novel Methods for Global Optimisation of Multiple Gravity Assist Trajectories

4.1 Introduction

Work Package 3 analysed the performance of a variety of modern global optimisation techniques on multiple gravity assist trajectories. Although there were some algorithms (most notably Differential Evolution [42]), that showed significantly better performance than the rest, it was concluded that the best way of developing a tool for optimising trajectories was to develop a method for utilising domain knowledge to prune the search space in order to yield a significant improvement in performance.

This work package considers the problem of multiple gravity assist trajectories with a known planetary sequence and no Deep Space Manoeuvres. In such cases, it has been shown that the vast majority of the search space consists of infeasible, or very undesirable, solutions. Therefore, rather than developing a better optimiser a method for producing reduced search spaces for pruning was created, thus allowing standard global optimisation techniques to be applied more successfully to the reduced box bounds.

The technique created is named the Gravity Assist Space Pruning (GASP), and is fully implemented in a Windows XP application created using Borland C++ Builder 5.

4.2 Gravity Assist Space Pruning Algorithm

This section describes the motivation behind and functionality of the GASP algorithm.

Consider the MGAproblem with a defined planetary sequence (e.g. Earth-Venus-Venus-Earth-Jupiter-Saturn) and no Deep Space Maneouvres. The decision vector for this problem is as follows

$$\mathbf{x} = \{t_0, t_1, t_2, t_3 ...\}, \tag{4.2.1}$$

where t_0 is the launch date, t_1 is the phase time from the first to second planet, t_2 from the second to third planet etc. A Lambert solver[1] is then used to calculate appropriate Keplerian orbits between the planetary positions in the given time, and then a powered swingby model is applied, such as that designed by Gobetz [15].

4.2.1 Single interplanetary transfer

Firstly, let us consider the simplest case of a single interplanetary transfer with a braking manoeuvre at the target planet.

The objective function assumed is a simple minimisation of total thrust (the sum of the initial hyperbolic excess velocity, v_i, and braking manoeuvre, v_f), so

$$f = v_i + v_f. \tag{4.2.2}$$

[1]unpublished work by Dario Izzo

The decision vector in the single transfer case will be $\mathbf{x} = \{t_0, t_1\}$. An important observation is that this search space will contain a line for each time t, that a probe can arrive at the final planet, such that $t_0 + t_1 = t$. Obviously, at a given time t, regardless of the launch time or departure time, the target planet will be *in the same position and have the same velocity*. Therefore, it is beneficial to consider the search space as t_0, $t_0 + t_1$ i.e. departure time at the first planet compared to arrival time at the second.

The optimisation method to be investigated is grid sampling. Grid sampling is usually considered a very inefficient optimiser, and this is true in high dimensionalities. For example, using the enumerative search in the Swingby Calculator application [7] yields optimisation times approaching an hour for relatively small search spaces (on a 600Mhz Pentium Processor). However, for only 1 and 2 dimensions grid sampling is relatively efficient, as long as the search space is reasonably smooth and the exact optimum is not required.

Therefore, the objective function for a single interplanetary transfer may be grid sampled at an appropriate resolution in the departure time vs arrival time domain efficiently, although in this case most other optimisation methods would undoubtedly yield better results in terms of objective function evaluations. However, the grid sampled version will require many less Ephemeris calculations, as the same positions/velocities need not be recalculated for a given departure or arrival time. If the 2D search space was discretised into x cells in each dimension, only $2x$ Ephemeris calculations are required for the entire sampling, and x^2 Lambert problem solutions. By comparison, 2 Ephemeris calculations would be required by each objective function evaluation in a standard optimiser.

Even in the single interplanetary transfer case, a large proportion of the search space corresponds to undesirable solutions i.e. those with impractical

C3. To illustrate this, the optimisation of an Earth-Mars transfer was considered between the dates (-1200 to 600 MJD2000) and phase times of 25 to 515 days. A sampling resolution of 10 days was used in both axes. Only 12.5% of this search space had a C3 of less than 25km^2/s^2. Figure 4.1 shows this search space plotted as departure time vs arrival time - the diagonal red lines delineate the sampled portion of the search space, and the black regions within the lines indicate trajectories with a C3 of greater than 25km^2/s^2.

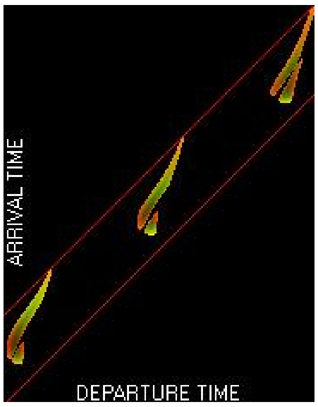

Figure 4.1: A grid sample Earth-Mars transfer. The black regions within the delineating red lines indicate solutions with a C3 of greater than 25km^2/s^2

Even allowing an enormous C3 of 100km^2/s^2, only 33% of the search space becomes valid. As a consequence, in gravity assist and multiple gravity assist cases starting with an Earth-Mars transfer in these bounds, at least 87.5% of the overall search space *must* correspond to undesirable solutions.

The GASP algorithm was design to efficiently detect and prune infeasible parts of the space, leaving several sets of box bounds with vastly smaller contents. These reduced box bounds may then be optimised efficiently using a standard optimisation method.

Hyperbolic Excess Velocity Constraint

The maximum allowable hyperbolic excess velocity is the first main constraint of the GASP algorithm, as it determines possible launch dates to the first target planet.

Braking Manoeuvre Constraint

As well as the C3 constraint, it is logical to add a constraint on the maximum braking manoeuvre that the spacecraft can perform. Applying a C3 constraint of 25km^2/s^2 and a braking manoeuvre constraint of 5km/s yields the search space shown in Figure 4.2. Less than 5% of the search space now yields feasible trajectories.

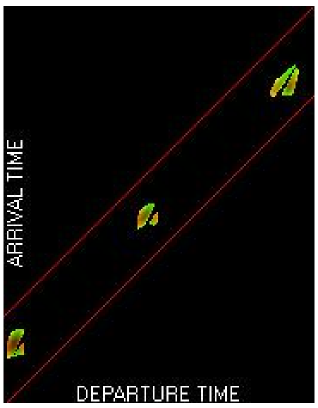

Figure 4.2: An Earth-Mars transfer with both C3 and braking manoeuvre constraints applied

By applying two very simple constraints to the interplanetary case it has been shown that a very significant reduction in search space can been achieved, leaving clear launch windows and arrival time windows.

4.2.2 Forward Constraining

It has been shown that the C3 and braking manoeuvre constraints alone significantly reduce the search space content for an interplanetary transfer. From Figure 4.2, it can be seen that many of the horizontal lines of pixels are completely black - this implies that it is *not possible* to arrive at the target planet on the corresponding date within the current constraints.

This information is the key to the functionality of the GASP algorithm: if no feasible trajectories arrive at a planet on a given date then there can be no departures from the planet on that date (assuming the change in velocity from the swingby is instantaneous).

Now consider a trajectory with a single gravity assist. Using grid sampling on this function would usually involve sampling in 3 dimensions, and hence as additional planets were added the number of objective function evaluations would increase exponentially. Instead, with GASP, the search space is sampled as a cascade of 2 dimensional search spaces, each with possible departure dates in the horizontal axis and prospective arrival dates in the vertical axis. Because of this, the number of Lambert problem evaluations is vastly reduced.

4.2.3 Gravity assist thrust constraint

Two constraints are added in order to maximise the probability of gravity assists being feasible. The first such constraint is the gravity assist thrust constraint, which limits the maximum absolute difference between incoming and outgoing velocities during a gravity assist to some threshold, T_v. This threshold is set separately for each gravity assist.

The following is then performed for *every* arrival time at a planet:

1. Calculate the bounds on incoming velocity, $v^i_{\min}$ and $v^i_{\max}$.

2. Invalidate any outgoing trajectories that do not have outgoing velocities in the range $[v^i_{\min} - T_v - L_v, v^i_{\max} + T_v + L_v]$, where L_v is an appropriate tolerance based on the Lipschitzian constant of the current phase plot.

3. Calculate the modified bounds on outgoing velocity, $v^f_{\min}$ and $v^f_{\max}$.

4. Invalidate any incoming trajectories with velocities outside the range $[v^f_{\min} - T_v - L_v, v^f_{\max} + T_v + L_v]$.

4.2.4 Gravity assist angular constraint

The gravity assist angular constraint removes infeasible swingbys from the search space on the basis of them being associated with a hyperbolic periapse under the minimum safe distance for the given gravity assist body. This is determined over every arrival date at a planet as follows, assuming x valid incoming trajectories and y valid outgoing trajectories:

```
1.  For all $x$ incoming trajectories
2.      For all $y$ incoming trajectories
3.          If the swingby is valid for the current incoming
            and outgoing trajectory, mark both incoming and
            outgoing trajectory as valid.
4.      End
5.  End
6.  Invalidate all trajectories not marked as valid
```

The swingby angle is decreased by an appropriate Lipschitzian tolerance θ_L, in order to compensate for the grid sampled nature of the search space.

4.3 Time and Space Complexity

This section determines the time and space complexity of the GASP algorithm. It will be shown that GASP scales quadratically in space and quartically in time with respect to the number of gravity assist manoeuvres considered. For simplicity, the following analysis assumed that the initial launch window and *all* phase times are the same.

4.3.1 Space Complexity

Consider a launch window of size discretised into x cells and a mission phase time also discretised into x cells. For the first phase x^2 Lambert problems must be sampled. The next phase will need to sample $(x+x)x = 2x^2$, as the number of possible times that the planet may be arrived at is doubled (minimum launch date, minimum phase time to maximum launch date, maximum phase time). The third phase will require $3x^2$ Lambert function evaluations, and the n^{th} phase nx^2. This gives the series

$$O(n) = x^2 + 2x^2 + 3x^2 + \ldots + nx^2 \tag{4.3.1}$$

$$O(n) = x^2(1 + 2 + 3 + \ldots + n) \tag{4.3.2}$$

$$O(n) = x^2 \frac{n(1+n)}{2}. \tag{4.3.3}$$

Therefore, the amount of space required for n phases is only of the order $O(n^2)$, rather than $O(x^n)$ for full grid sampling.

Similarly, it is clear that the space complexity with respect to the resolution x, is also of the order $O(x^2)$.

4.3.2 Time Complexity

The memory space required 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 GASP algorithm must also be of the order $O(n^2)$.

Launch energy constraint complexity

The launch energy constraint is only applied in the first phase, and hence is independent of the number of swingbys. The time complexity is $O(x^2)$ with respect to resolution.

Gravity assist thrust constraint complexity

The time complexity of applying the gravity assist thrust constraint is $O(n^2)$ with respect to dimensionality, due to the inevitable increase in size of later phase plots to encompass all possible arrival dates.

The first phase requires operations of the order $2x \times (x + 3x)$ in order to perform the constraining of outgoing velocity from incoming velocity (the back constraining may be ignored at this point). The second phase requires operations of the order $3x \times (2x + 4x)$. In general, the n^{th} phase requires operations of the order $2n^2x^3$. Therefore, the total number of operations over all phases is

$$2x^2[2^2 + 3^2 + 4^2 + \ldots + n^2] = 2x^2\frac{n(n+1)(2n+1)}{3} \tag{4.3.4}$$

Therefore, applying this constraint yields cubic time complexity in dimensionality and quadratic complexity in resolution.

Gravity assist angular constraint complexity

The maximum number of swingby models that must be calculated for the first phase is close to $x \times 2x \times 3x = 6x^3$. For the second swingby, this is $2x \times 3x \times 4x = 24x^3$. In general, for n phases, the upper bound on the number of swingby calculations, α, is

$$\alpha = 3.2.1.x^3 + 4.3.2.x^3 + 5.4.3.x^3 + \ldots + (n+2)(n+1)nx^3 \qquad (4.3.5)$$

From [40], it can be shown that the total number of these operations must be

$$\alpha = x^3 \sum_{1}^{n} (n+2)(n+1)n = \frac{n(n+1)(n+2)(n+3)}{4}. \qquad (4.3.6)$$

Therefore, the overall time complexity with respect to resolution is $O(x^3)$, while the time complexity with respect to dimensionality is $O(n^4)$. Therefore, the gravity assist angular constraint is the most computationally expensive and hence is applied after GA thrust constraint in order to minimise the number of swingby models that must be calculated.

Overall time complexity

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

4.4 Results

This section demonstrates the improvements that GASP can make over Differential Evolution alone and compares GASP with the grid sampling option of Swingby Calculator.

4.4.1 Earth-Venus-Mars-Earth

The first trajectory to be considered is EVME, culminating in a flyby of Earth. This trajectory was considered in [7] as a test case for the Swingby Calculator application developed by JAQAR Space Engineering. The bounds chosen were as follows:

- $t_0 \in [3000, 4000]$ MJD2000
- $t_1 \in [14, 494]$ days
- $t_2 \in [21, 491]$ days
- $t_3 \in [25, 495]$ days

The above bounds are significantly more relaxed than those used when validating swingby calculator - the optimisation in that case took 27 minutes with a sample resolution of 7 days, and 7311616 trajectory possibilities were considered using grid sampling. Using GASP, in total only 45330 Lambert problems were solved and 2074 Ephemeris calculations required: the sample resolution was also higher at 5 days. Altogether, this took less than a second on a 2.8GHz Pentium 4 processor. Applying differential evolution (40 individuals, 2000 iterations) to the located solution families took several more seconds, and the best found trajectory compares closely to that presented in [7] (see Figure 4.3).

To determine the reliability of the bounds created by GASP, differential evolution was applied to the reduced bounds over 20 trials. 14 out of these converged to the best known minimum (5044m/s), and 6 to the second best minimum (5058m/s, which only requires 14m/s additional thrust). However, when applying differential evolution alone over the full problem domain bounds, only 9 out of 20 trials converged to the global minimum, 3 to the second best minimum and the rest on significantly poorer solutions.

These results confirm that the pruned spaces produced by GASP are indeed valid, in that they will encompass the global optimum if the thresholds are appropriately chosen: differential evolution over the entire domain did not find any superior solutions. Also, the level of success of DE alone on this problem indicates that several restarts would be required in order to

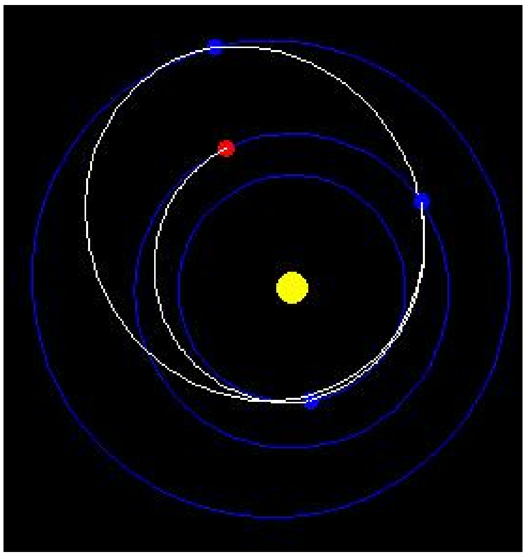
Figure 4.3: An optimal Earth-Venus-Mars-Earth trajectory with flyby

guarantee the global optimum - GASP, by comparison, had a 100% success rate in finding one of the two best solutions, and application of the GASP algorithm was significantly less computationally expensive than a single differential evolution trial (which requires 240000 Lambert solutions and 320000 Ephemeris calculations)

4.4.2 Earth-Venus-Venus-Earth-Jupiter-Saturn transfer (short launch window)

This section considers the optimisation of an EVVEJS transfer with an orbital insertion, where the objective function is the minimisation of the sum of the launcher and probe thrust. The bounds on the decision vector were as follows:

- $t_0 \in [-1200, 600]$ MJD2000
- $t_1 \in [14, 284]$ days

- $t_2 \in [22, 442]$ days
- $t_3 \in [14, 284]$ days
- $t_4 \in [99, 1989]$ days
- $t_5 \in [366, 7316]$ days

GASP was applied to this problem with a sampling resolution of 10 days. In order to complete the sampling, 144498 Lambert problem solutions were required, and 3749 Ephemeris calculations.

The following constraints were defined in the GASP algorithm:

- $T_{HEV} = 8000m/s$
- $T_{GA_{1,...,4}} = 1000m/s$
- $T_{\text{Brake}} = 5000m/s$

This configuration yields two major solution families, one with a launch window of -920 to -660MJD2000, and the other 280 to 490MJD2000. Differential Evolution was applied to the accumulation of each solution family (the tightest decision vector bounds that all solution family nodes exist within). A population of 40 individuals was used and a terminal number of 2000 iterations were allowed. Note that this corresponds to $40 \times 2000 \times 5 = 400000$ Lambert problem solutions and 480000 Ephemeris calculations.

The later launch window was eliminated immediately as applying Differential Evolution did not yield any valid solutions. Further optimisation on this launch window has consistently optimised to the same invalid minima.

The earlier launch window proved much more promising and, as a consequence, 20 optimisation trials were performed. Of these trials, 19 found the second best known optima to this problem (5225m/s) to within 1m/s, and one came close to the best known optima, (4870m/s). Experimentation has shown this minimum has a very small basin of attraction with respect to this objective function, and is exceptionally hard to find even in the reduced search space.

When applying Differential Evolution alone to the entire search space, only 7 out of the 20 trials found the second best minimum, and none the best known. Using GASP, it is apparent that there is an extremely high probability that at least the second best solution will be found.

The objective function was then altered to penalise any trajectory with hyperbolic excess velocity of greater than 3000m/s, and the trials run again. Now only 2 of 20 of the GASP constrained trials failed to find the best known solution to within 10m/s (instead finding the second best one), while again 7 out of 20 optimisation of the entire domain found the second best solution, and none located the best.

Again, these results highlight the added power of using the GASP algorithm. Not only does it allow effective visualisation of the search space, but it drastically reduces the requirement for optimisation restarts in order to find good solutions, and at a fraction of the computational expense of an optimisation restart.

4.4.3 Earth-Venus-Venus-Earth-Jupiter-Saturn transfer (long launch window)

The above optimisation problem was retested with identical parameters apart from the initial launch window was 10000 days (over 27 years) rather than 1800. Applying GASP to this search space required 496865 Lambert problem solutions, and 11949 Ephemeris calculations. 111 Megabytes of memory were allocated for the grid sampled tables. The C3 constraint was here lowered from 8000m/s to 5500m/s, reducing the number of solution families to only two - these corresponded very closely to the solution families located for the short launch window.

Applying Differential Evolution alone to the entire domain in 20 trials (with the same parameters) yielded only 4 trials that came close to the second best minima. When Differential Evolution was applied the GASP defined bounds, all 20 trials located the second best known minima.

Therefore, although GASP does not guarantee that the global optimum can be located on this problem, it certainly increases the probability of locating good solutions.

4.5 Conclusions

This report has described the Gravity Assist Space Pruning algorithm, proved that it has both polynomial time complexity and space complexity, and furthermore demonstrated that it produces significant benefits over optimising the entire domain with relatively little computational expense. Additionally, the GASP algorithm allows intuitive visualisation of a high dimensional search space, and facilitates the identification of launch windows and different mission options.

References

[1] Baker, C.A., Watson,L.T., Grossman, B., Mason, W.H. and Haftka, R.T. 'Parallel Global Aircraft Configuration Design Space Exploration', International Journal of Computer Research 10(4), pp. 79-96,2001.

[2] Baker, R.M.L., and Makemson, M.W. 'An Introduction to Astrodynamics'. New York and London, Academic Press, 1960.

[3] Bate, R. R., Mueller, D. D., White, J. E., 'Fundamentals of Astrodynamics'. Dover, New York, 1971.

[4] Belbruno, E.A., Carrico, J.P., 'Calculation of Weak Stability Boundary Ballistic Lunar Transfer Trajectories.' AIAA Paper 2000-4142, AIAA/AAS Astrodynamics Specialist Conference, August 2000.

[5] Betts, J.T. 'Practical methods for optimal control using nonlinear programming.' SIAM, Philadelphia, 2001.

[6] Biesbroek, R., Ockels, W., and Janin, G. 'Optimisation of Weak Stability Boundary Transfers from GTO to the Moon using Genetic Algorithms', IAF-99-A.6.10, Amsterdam-Netherlands, 4-8 Oct 1999.

[7] Biesbroek, R., Ancarola, B. 'Optimisation of Launcher Performance and Interplanetary Trajectories for Pre-Assessment Studies', IAF abstracts,

34th COSPAR Scientific Assembly, The Second World Space Congress, held 10-19 October, 2002 in Houston, TX, USA., p.A-6-07

[8] Blackwell, T.M. and Branke, J., 'Multi-Swarm Optimisation of the Moving Peaks Function'. Applications of Evolutionary Computing EuroWorkshops 2004, Proceedings, LNCS 3005, Springer-Verlag (2004) pp. 489-500.

[9] Boeing, 'Fact Sheet - Xenon Ion Propulsion', http://www.boeing.com/defense-space/space/bss/factsheets/xips/xips.html

[10] Chiaradia, A.P.M., Kuga, H.K., Prado, A.F.B.A. 'Autonomous Coplanar Orbital Maneuver Using GPS System'. In: 53Rd. International Astronautical Congress, 2002.

[11] Debban, T.J., McConaghy, T.T., Longuski, J.M., 'Design and Optimization of Low-Thrust Gravity-Assist Trajectories to Selected Planets.' AIAA 2002-4729, Astrodynamic Specialist Conference and Exhibit, 2002.

[12] Dixon, L.C.W. and Szego, G.P. 'The optimization problem: An introduction' in Dixon, L.C.W. and Szego, G.P. (Eds.), 'Towards Global Optimization II', New York: North Holland, 1978.

[13] Gage P.J., Braun R.D., Kroo I.M. Interplanetary Trajectory Optimisation Using a Genetic Algorithm. The Journal of the Astronautical Sciences, Vol. 43, No. 1, January-March 1995, pp. 59-75.

[14] Garwin, R.L., 'Solar Sailing - A Practical Method of Propulsion Within the Solar System' Jet Propulsion 28:3 pp. 188-190, 1958.

[15] Gobetz, F.B., 'Optimal transfer between hyperbolic asymptotes', AIAA J. 1(9), 1963.

[16] Goldstein, A.A. and Price, I.F., 'On descent from local minima' Math. Comput., Vol. 25, No. 115, 1971.

[17] Hall, C.D. and Ross, I.D. 'Optimal Attitude Control for Coplanar Orbit Phasing Transfers' Advances in the Astronautical Sciences (115), pp. 79-94, 2003.

[18] Hargraves, C., and Paris, S. 'Direct Trajectory Optimization Using Nonlinear Programming and Collocation' AIAA J. Guidance and Control (10), pp. 338-342, 1987.

[19] Hartmann, J.W., Coverstone-Carroll V.L., Williams S.N., 'Optimal Interplanetary Spacecraft Trajectories via a Pareto Genetic Algorithm', Journal of the Astronautical Sciences 46(3) pp. 267-282, 1998.

[20] Hohmann, W., 'Die erreichbarkeit der himmelskorper', Oldenbourg, Munique.

[21] Holland, J.H. 'Adaptation in Natural and Artificial Systems', University of Michigan Press, Ann Arbor, 1975.

[22] Huyer, W. 'A comparison of some algorithms for bound constrained global optimization', WWW-document, http://www.mat.univie.ac.at/neum/glopt/contrib/compbound.pdf, 2004.

[23] Huyer, W., and Neumaier, A., 'Global optimization by multilevel coordinate search', J. Global Optimization 14, pp.331-355, 1999.

[24] Izzo D., Bevilacqua R., Valente C. 'Optimal large reorientation manoeuvre of a spinning gyrostat', 6th International Conference On Dynamics and Control of Systems and Structures in Space, Riomaggiore, Italy, 2004.

[25] Jones, D.R., Perttunen, C.D. and Stuckman, B.E., 'Lipschitzian optimization without the Lipschitz constant', J. Optimization Th. Appl. 79, pp. 157-181, 1993.

[26] Kennedy, J., Eberhart, R., 'Particle swarm optimization', Proc. IEEE Int. Conf. on Neural Networks, pp. 1942-1948, 1995.

[27] Koon W.S., M.W. Lo, J.E. Marsden and S.D. Ross. 'Heteroclinic Connections between Lyapunov Orbits and Resonance Transitions in Celestial Mechanics Ballistic Capture'. Chaos 10: 427-469, 2000.

[28] Koon, W.S., Lo, M.W., Marsden, J.E. and Ross, S.D. 'Low Energy Transfer to the Moon.' Celestial Mechanics and Dynamical Astronomy 81, pp. 63-73,2001.

[29] Kroese, D.P., and Rubinstein, R.Y., 'The Cross-Entropy Method: A Unified Approach to Combinatorial Optimization', Monte-Carlo Simulation and Machine Learning. Springer, 2004.

[30] Labunsky, A.V., Papkov, O.V. and Sukhanov, K.G. 'Multiple Gravity Assist Interplanetary Trajectories' Gordon and Breach Science Publishers, ISBN 90-5699-090-X, 1998.

[31] McConaghy, T.T., Debban, T.J., Petropoulos, A.E., and Longuski, J.M., 'An Approach to Design and Optimization of Low-Thrust Trajectories with Gravity Assists' AAS/AIAA Astrodynamics Specialist Conference, AAS Paper 01-468, Quebec City, QC, Canada, July-August 2001.

[32] Muhlenbein, H., Mahnig, T. and Rodriguez, O. 'Schemata, Distributions and Graphical Models in Evolutionary Optimization' Journal of Heuristics (5) pp. 215-246, 1999.

[33] Neumaier, A., 'Complete Search in Continuous Global Optimization and Constraint Satisfaction', 2003.

[34] Pardalos, P.M. 'Recent Advances in Global Optimization', 1999.

[35] Petropoulos, A. E., Longuski, J. M., and Vinh, N. X. 'Shape-Based Analytic Representations of Low-Thrust Trajectories for Gravity-Assist Applications' AAS/AIAA Astrodynamics Specialist Conference, AAS Paper 99-337, Girdwood, Alaska, Aug. 1999.

[36] Prado, A.F.B.A. 'Optimal Transfer and Swing-By Orbits in the Two- and Three-Body Problems.' PhD Thesis, University of Texas, 1993.

[37] Prado, A.F.B.A. and Broucke, R.A. 'The Minimum Delta-V Lambert's Problem' Proceedings of the VII Colquio Brasileiro de Dinmica Orbital, pp. 16, Santos, Brazil.

[38] Press, W.H., Flannery, B.P., Teukolsky, S.A., and Vetterling, W.T. 'Runge-Kutta Method' and 'Adaptive Step Size Control for Runge-Kutta.' 16.1 and 16.2 in Numerical Recipes in FORTRAN: The Art of Scientific Computing, 2nd ed. Cambridge, England: Cambridge University Press, pp. 704-716, 1992.

[39] Raphael, B., and Smith, I.F.C., 'A direct stochastic algorithm for global search', J. of Applied Mathematics and Computation, Vol 146, No 2-3, pp. 729-758, 2003.

[40] Riordan, J., 'Combinatorial Identities', Wiley, 1968, p. 77.

[41] Sohn, R.L., 'Attitude Stabilization by Means of Solar Radiation Pressure' ARS Journal 29, pp. 371-373, 1959.

[42] Storn, R. and Price, K., 'Differential Evolution - a Simple and Efficient Heuristic for Global Optimization over Continuous Spaces', J. of Global Optimization 11, pp.341-359, 1997.

[43] Törn, A. and Žilinskas, A. 'Global Optimization', Lecture Notes in Computer Science 350, Springer-Verlag, pp. 255, 1989.

[44] Törn, A., M. Ali and S. Viitanen, 'Stochastic global optimization: Problem classes and solution techniques', Journal of Global Optimization 14, pp. 437-447, 1999.

[45] Vasile M., Campagnola S., Bernelli-Zazzera F. Electric Propulsion Options for a Probe to Europa. International Symposium on Low-thrust trajectories, Tolouse 18-20 June 2002.

[46] Vasile M. 'Robust Optimisation of Trajectories Intercepting Dangerous NEO'.AIAA-2002-4719.AIAA/AAS Astrodynamic Specialist Conference, Monterey,California 5-8 August 2002.

[47] Vasile, M. 'A Systematic-Heuristic Approach for Space Trajectory Design.' New Trends in Astrodynamics and Applications, Washington, pp.20-22, 2003.

[48] Vasile, M., Galvez, A., Summerer, L., Ongaro, F. 'Design of Low-Thrust Trajectories for the exploration of the Outer Solar System'. IAC-03/A.P.14, 2003

[49] Weisstein, E.W. 'B-Spline' From MathWorld–A Wolfram Web Resource. http://mathworld.wolfram.com/B-Spline.html

[50] Wolfram, S., 'The Mathematica Book, Fifth Edition', Wolfram Media, 2003.

[51] Yokoyama, N. ‘Trajectory Optimization of Space Plane Using Genetic Algorithm Combined With Gradient Method’, Proc. of 23rd International Congress of Aerospace Sciences, 2002.

Appendix A - Taxonomy Summary Tables

Tables 2 to 6 illustrate the use of the mission design problem taxonomy presented in this report to define specific mission design problems. Each table represents choices along one dimension of the taxonomy. By following the instructions in the captions, the reader can characterise different mission design problems by working through Tables 4.1 to 4.5.

Table 4.1: Checkbox for problem constraints - check one in each row

Variable	**U**	**#**	**B**	$f(t)$	**O**
$\mathbf{x}_i$					N/A
$\dot{\mathbf{x}}_i$					N/A
t_i				N/A	N/A
$\mathbf{x}_f$					
$\dot{\mathbf{x}}_f$					
t_f				N/A	N/A
Additional Constraints					

Table 4.2: Summary of control classes - Select one. Later options indicate increased optimisational complexity.

Control Class	**Subclass**	**Check/Value**
Impulsive	Fixed n-Impulsive	
Continuous	Simple	
Impulsive	Variable Impulsive	
Continuous	Complex	

Table 4.3: Decision variables/Objectives - check one or more

Quantity	**Decision Variable**	**Objective**
Thrust		
Mission time		
Velocity		
Angular momentum		
Other variables		

Table 4.4: Spacecraft model complexity - check one. Later options indicate increased optimisational complexity.

Model Complexity	**Check**
Point-mass	
Rigid body	
Low level control	

Table 4.5: Astrodynamic model - check applicable and add appropriate values. Optimisational complexity increases with each checked item and increased n body dynamics

Model Complexity	**Check/Value**
n-body dynamics (2+)	
Non-planar model	
Elliptic orbits	
Other factors	