Trajectory Optimization Using Global Methods IEPC05-173

Michael A. Paluszek[*] and Stephanie J. Thomas[†]

Princeton Satellite Systems, Princeton, NJ

This paper compares three different global indirect approaches for solving the problem of finding optimal trajectories for low thrust spacecraft. The three methods tested were downhill simplex, genetic algorithms and simulated annealing. Two problems are analyzed. The first is a planar low thrust problem. This is the problem of finding a minimum time trajectory from Earth to Mars orbit using a fixed thrust electric propulsion system. The second problem is the 3 dimensional problem of orbit transfer into an inclined heliocentric orbit.

Global optimization methods have the advantage of not requiring a good initial guess for the costates, or for the controls, and having some ability to avoid falling into local minima. In addition because they employ the costate equations, only a small number of parameters are needed to determine the trajectory.

This paper explores the use of global methods and introduces a methodology for evaluating global methods for use in optimization. Details such as the creation of scalar cost functionals and termination conditions for numerical integration are discussed. The results show that with a reasonable amount of work by the analyst all of the global methods employed in this paper can be used successfully to obtain solutions to 2D and 3D trajectory optimization problems.

I. Introduction

TRAJECTORY optimization has been studied since the 1950's in the context of low-thrust propulsion in space. Bryson,[1] Edelbaum,[2] Gath,[3] Kim,[4] Whiting,[5] and Betts[6] provide comprehensive literature reviews of trajectory optimization.

In the most general problem we would like to start in any orbit, including an orbit about a departure planet, and proceed to any other orbit, including an orbit about a target planet while minimizing some weighted combination of fuel and time while at the same time satisfying some set of path and thermal constraints and handling limitations on power. The trajectories, if interplanetary, may also involve gravity maneuvers.[7] The available controls might be thrust direction, thrust magnitude and exhaust velocity. The trajectory problem might involve multiple stages where the point of transition from one stage to another would also be part of the solution. The method would find the global minimum and not require the person operating the optimizer to be an optimization expert. The ideal method should be amendable to online autonomous operation since it would be necessary, for an autonomous vehicle, to correct the trajectory due to model errors and unmodeled disturbances and so the method may need to be integrated into a guidance and navigation system.[8]

Optimization methods can be broadly classified into direct and indirect methods. Direct methods find the optimal control directly and employ only the dynamical equations and constraint equations. Nonlinear programming[9] and evolutionary methods[10] have been used to solve trajectory optimization problems by the direct method. Indirect methods solve for the costates of the systems, that is the Lagrange multipliers for the system, and from the costates derive the controls. Indirect methods require both the dynamical and costate equations to be solved simultaneously. Many methods for solving the indirect method have been studied including gradient methods,[5] simulated annealing[11] and genetic algorithms.[12]

[*]President, Princeton Satellite Systems, 33 Witherspoon Street, Princeton, New Jersey, 08542, USA, map@psatellite.com
[†]Senior Technical Staff, Princeton Satellite Systems, sjthomas@psatellite.com

This paper will compare three global optimization methods, genetic algorithms, simulated annealing and downhill simplex as applied to 2D and 3D single stage low-thrust trajectory optimization. Both problems assume constant thrust but time varying mass. The paper first looks at the general optimization problem and then presents results about global optimization methods using tests for such algorithms. A simpler trajectory problem, Zermelo's problem, is studied to give insight into the behavior of global optimization tools. The 2D problem is solved using all three methods for an Earth to Mars trajectory problem. Two of the methods are applied to a 3D problem of transfer into a high inclination orbit.

II. Formulation of the Problem

The general problem for a single stage optimization problem is to minimize the cost function

$$J = \phi(x(t_f), t_f) + \int_{t_0}^{t_f} L(x, u, t)dt \tag{1}$$

subject to the dynamical equations

$$\dot{x} = f(x, u, t) \tag{2}$$

where $\phi(x(t_f), t_f)$ is a cost associated with the final state, x and time, t. $L(x, u, t)$ is the cost associated with the state, time and control, u as the trajectory progresses. For the minimum time problem, $L = 1$.

The dynamical equations are then appended to the cost functional by the Lagrange Multipliers, λ, which are functions of the states and time and are also known as the costates. The actual cost does not change since by definition the function multiplied by the costates is always zero. Additional constraints can be added to the cost functional as needed by including one additional costate per constraint.

$$J = \phi(x(t_f), t_f) + \int_{t_0}^{t_f} L(x, u, t)dt + \lambda^T \left(f(x, u, t) - \dot{x}\right) \tag{3}$$

The Hamiltonian is

$$H(x, u, t) = 1 + \lambda^T f(x, u, t) \tag{4}$$

The solution of the optimal control problem requires the solution of the following equations

$$\dot{\lambda}^T = -\frac{\partial H}{\partial x} \tag{5}$$

$$\frac{\partial H}{\partial u} = 0 \tag{6}$$

$$\dot{x} = f(x, u, t) \tag{7}$$

If the final time is not constrained and the Hamiltonian is not an explicit function of time, then

$$H(t) = 0 \tag{8}$$

We seek to solve the following boundary value problem. The state equations are

$$\dot{x} = f(x, u, t) \tag{9}$$

Boundary conditions are set for the state equations. The costate equations are

$$\dot{\lambda} = -f_x^T \lambda \tag{10}$$

where the subscript denotes the partial with respect to the vector x and their boundary conditions are unknown. The optimality condition is

$$0 = f_u^T \lambda \tag{11}$$

where the subscript denotes the partial with respect to the control vector u which provides a relationship between the controls and the costates.

The boundary conditions may be on the states or the costates. In the problems discussed in this paper the initial conditions are always known. However, not all end conditions may be specified. For those that are not specified the boundary condition on the costate is set to zero. If S is the set of specified terminal boundary conditions then

$$\begin{aligned} x_k(t_f) &= x_k, k \in S && (12) \\ \lambda_k(t_f) &= 0, k \ni S && (13) \end{aligned}$$

III. Global Methods

A. Downhill Simplex

The downhill simplex method is due to Nelder and Meade.[13] A simplex is defined as a figure of $N+1$ vertices in an N-dimensional space that do not lie in a hyperplane, i.e. have a finite volume. This would be a tetrahedron in 3 dimensional space. Each simplex is a solution in the search space. The simplex can be expanded

$$x = x_0 + \lambda \tag{14}$$

contracted

$$x = x_0 - \lambda \tag{15}$$

or reflected

$$x = -x_0 \tag{16}$$

Downhill simplex first finds the points where the objective function is the highest moving the opposite face of the simplex to a lower point. These steps are called reflections. Steps are taken to maintain the volume of the simplex. The method tries to take larger steps whenever possible. When it reaches a "valley floor" it contracts itself in a transverse direction and tries to move down the valley.

This method requires only function evaluations, derivatives of the functions are not required. This is particularly useful when derivatives are not available or are difficult to compute.

B. Genetic Algorithms

Genetic algorithms are a subset of evolutionary algorithms.[14] A genetic algorithm is an iterative procedure that consists of a constant size population of individuals, each one representing a solution in a given problem space. The algorithm is applied to spaces too large to be solved by exhaustive searching.

During each generation, individuals are chosen to reproduce. There are many methods for creating children but generally they involve crossovers, requiring two parents, and mutations, requiring a single parent. A selection function then determines which individuals survive to the next generation. The functions may have some dependence on the fitness of the individuals or may be completely random. Elitist implementations always keep the fittest individual (or lowest cost solution, in this case) during selection. The size of the population and number of generations are important parameters in the algorithm.

Genetic algorithms are stochastic iterative processes that are not guaranteed to converge. The termination criteria must be set at some fitness level or a fixed number of generations.

Throughout this paper 20 to 30 generations and 20 to 30 individuals per generation are used for the genetic algorithms. It is not unusual for hundreds of individuals to be used per generation but also for solutions to require as few as 5 generations.

C. Simulated Annealing

Simulated annealing[15,16] is a global optimization method based on the an analogy with thermodynamics. At high temperatures the molecules of a liquid move freely with respect to each other. If the liquid is cooled slowly the thermal mobility is diminished and the atoms often can line themselves up in ordered lattices that are billions of atoms in 3 dimensions. This is a minimum energy state and if the cooling process is slow enough the crystal avoids falling into local minimums.

When minimizing a function, any downhill step is accepted and the process repeats from this new point. An uphill step may be accepted using the Metropolis criteria as the basis for decision. Thus simulated annealing process can escape from a local minimum. At each temperature the process is allowed to come to a new equilibrium.

The Metropolis criteria is an exponential function. It is computed for a given temperature. A random number is generated and compared against the value of the exponential.

As the optimization process proceeds, and temperature decreases, the length of the steps decline and the algorithm closes in on the global optimum.

D. Method Implementations

The particular methods use in this paper are fmins and fminsearch[17] for downhill simplex, GAOT[18] for genetic algorithms and Goffe's algorithm[15] for simulated annealing. In addition, a genetic algorithm, written based on the description given by Sipper[14] was used in the tests given in the next section to demonstrate that even an ad hoc genetic algorithm can produce quite reasonable results.

IV. Tests of Global Methods

Four optimization algorithms were tested using the De Jong functions. These five functions are a standard set used to test genetic algorithms. The five functions are illustrated in Figure 1. These functions illustrate situations that might be encountered in global optimization.

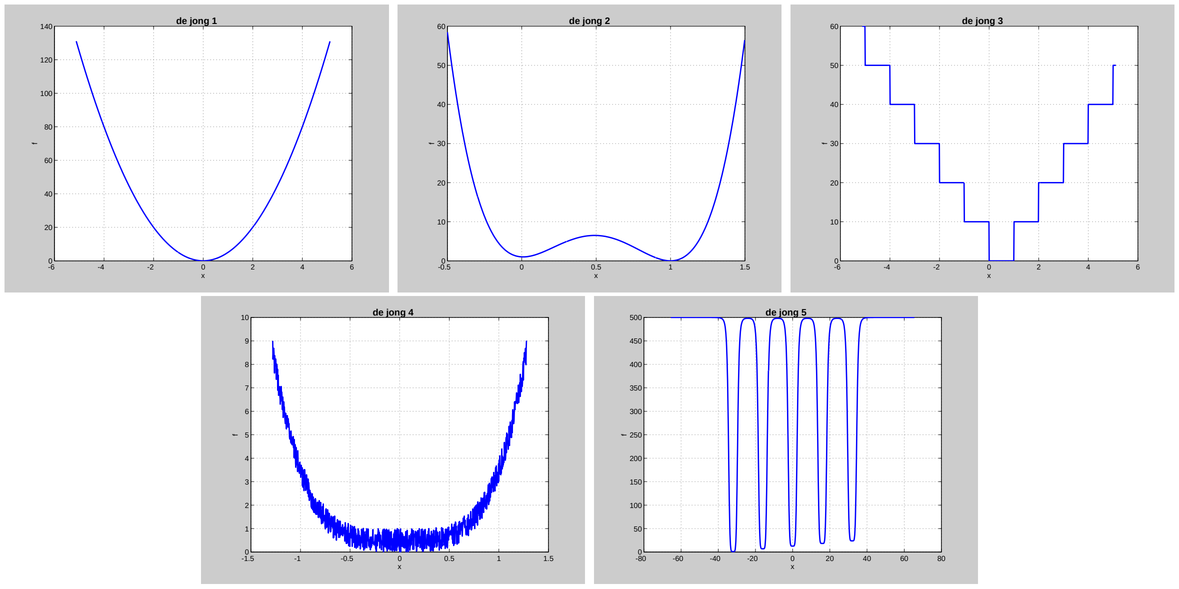


Figure 1. De Jong test cases

In each case all of the optimization variables were set equal to the same number within the ranges specified by De Jong. The first function tests the ability of the algorithm to find a simple minima. In the second, also known as Rosenbrock's function, has a shallow local minimum at zero and the true minimum at 1.0.

The next function is a stair case. The horizontal parts of the step can trap a function that interprets a non changing f as a minimum. In addition the cost function is discontinuous which would rule out the use of optimizers that require the cost to be a smooth function of the state.

The fourth function tests the effectiveness of the algorithm with noise. The true minimum is zero.

The fifth function has a series of local minima. A truly global algorithm will not get caught in any of the local minima. Unlike Rosenbrock's function the local minima are very deep.

Table 1 gives the final cost achieved by each algorithm and a reference value from the literature. The cost can be compared with the plots in Figure 1.

Table 1: Final cost for the De Jong test functions

Test	Ref	Downhill Simplex	Simulated Annealing	Genetic Algorithm	GAOT
de jong 1	0.0001	0	0	2.017e-07	2.017e-07
de jong 2	0.0001	1.865e-09	1.99e-08	1.509e-11	1.509e-11
de jong 3	0	0	0	0	0
de jong 4	0.0001	0.5174	0.1927	0.1767	0.5607
de jong 5	0.998	12.67	12.67	0.998	0.998

The genetic algorithms successfully solve De Jong 5. Simulated annealing and downhill simplex get stuck in a local optimum. Table 2 gives the wall clock time for each test. The tests were run in MATLAB on an Apple PowerBook G4. Table 3 gives the floating point operations for each test.

Table 2: Clock time

Test	Downhill Simplex	Simulated Annealing	Genetic Algorithm	GAOT
de jong 1	0.1914	4.14	7.372	4.945
de jong 2	0.02469	0.8763	6.494	4.555
de jong 3	0.003493	2.343	6.225	6
de jong 4	1.668	63.54	6.788	7.182
de jong 5	0.1531	4.981	37.25	25.13

Table 3: Floating point operations

Test	Downhill Simplex	Simulated Annealing	Genetic Algorithm	GAOT
de jong 1	1152	2.137e+05	8.909e+05	8.009e+05
de jong 2	3480	6.265e+04	7.274e+05	6.644e+05
de jong 3	513	5.55e+04	7.791e+05	7.341e+05
de jong 4	1.143e+06	1.288e+06	2.491e+06	1.798e+06
de jong 5	1.555e+04	8.041e+05	5.137e+06	3.319e+06

The clock time does not always correlate with the number of floating point operations. The clock time is effected by the number of `for` loops in the code (which slows things down considerably in MATLAB) and also what other processes are running on the computer at the same time. These timings are based on a single run of the algorithm.

V. Structure of the Optimizer

The structure of global optimization algorithms is straightforward. The optimizer guesses the initial costates. The state and costates are integrated by an integrator until a stopping condition is attained. The control is computed from the costates using an algebraic relationship. This may be analytical or it may be necessary to compute the control numerically. When the stopping condition is achieved the final states are input into the cost calculation. The cost calculation matches the final states against the desired states. It also computes any transversality conditions. The cost is passed back to the optimizer which creates a new set of initial costates. The structure of the optimizer is shown in Figure 2.

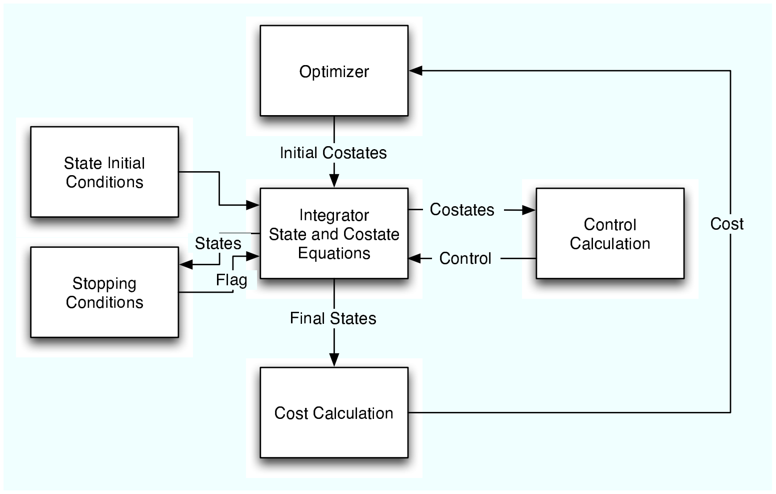


Figure 2. Structure of the optimizer

VI. Zermelo's Problem

Insight into issues of global optimization can be obtained by examining Zermelo's problem. Zermelo's problem is a 2D trajectory problem of a vehicle at constant velocity in a velocity field in which the velocity is a function of position. An analytical solution is possible for the case

$$\begin{aligned} \dot{x} &= V\cos\theta - V\frac{y}{h} \\ \dot{y} &= V\sin\theta \end{aligned} \tag{17}$$

where only the velocity in the x-direction is non-zero and it is a linear function of the y position of the vehicle. V is the velocity of the vehicle which is constant and θ is the angle of the vehicle relative to the x-axis and is the control in the problem.

The corresponding costate equations are

$$\begin{aligned} \dot{\lambda}_x &= 0 \\ \dot{\lambda}_y &= \lambda_x\frac{V}{h} \end{aligned} \tag{18}$$

λ_x is a constant and λ_y is

$$\lambda_y(t) = \lambda_x\frac{V}{h}t + \lambda_y(0) \tag{19}$$

The control angle θ is

$$\tan\theta = \frac{\lambda_y}{\lambda_x} \tag{20}$$

The solutions for the positions are

$$\begin{aligned} \frac{y}{h} &= \sec\theta - \sec\theta_f \\ \frac{x}{h} &= \frac{1}{2}\left[\sec\theta_f(\tan\theta_f - \tan\theta) - \tan\theta_f(\sec\theta_f - \sec\theta) + \log\frac{\tan\theta_f + \sec\theta_f}{\tan\theta + \sec\theta}\right] \end{aligned} \tag{21}$$

where θ_f is the final control angle and log is $\log_e$.

Figure 3 shows the resulting trajectory for the analytical solution to Zermelo's problem which is $\lambda = [-0.5; 1.866025]$. The arrows indicate the velocity vector.

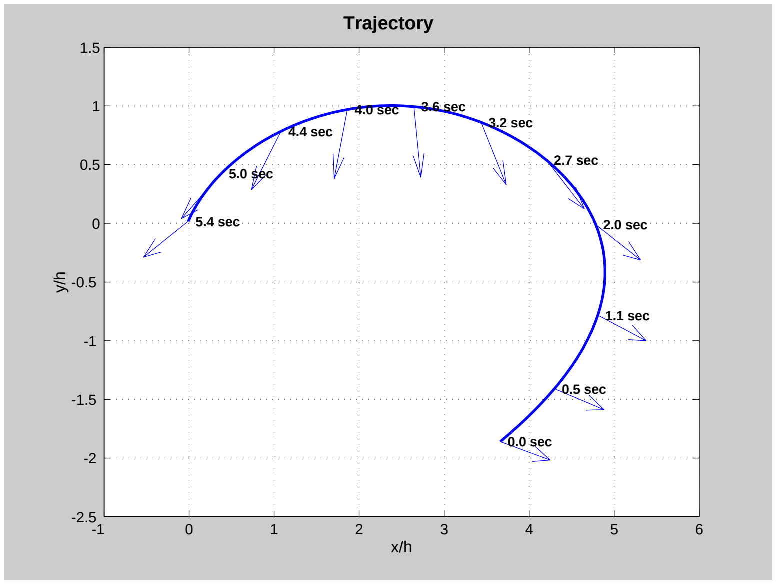


Figure 3. Analytical solution to Zermelo's Problem

For each method being tested the optimization parameters have been chosen, by a certain amount of trial and error, to get the best results. The final λ vector is different in each case. However, the control is determined by the ratio so the absolute values are not important. Table 4 gives the analytical and numerical solutions for the problem.

Table 4: Solutions

Test	λ_x	λ_y	Ratio
Analytical	-0.5	1.866025	-0.26795
Simplex	-0.65946	2.9404	-0.22428
Simulated Annealing	-0.68652	2.4593	-0.27915
Genetic Algorithm	-0.78899	2.9404	-0.26833

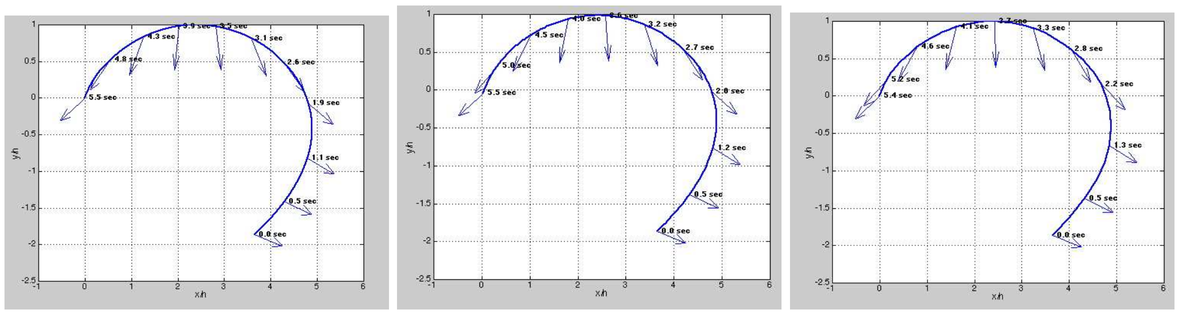


Figure 4. Numerical solutions to Zermelo's Problem (downhill simplex, simulated annealing and genetic algorithm))

None of the numerical methods converge on the exact solution. In each case, they fall into a suboptimal local minima. The trajectories all look similar to the analytical solution. The travel time for the genetic algorithm matches the analytical solution of 5.4 sec. The other two methods reach the terminal point in 5.5 sec, only 0.1 second longer.

It is interesting to look at the cost function versus costate. The cost function is the distance of the final trajectory point from the origin. Since this is a two dimensional problem the cost can be easily plotted (see Figure 5). This plot is limited to ±3 for both λ_x and λ_y. The red line shows the analytically obtained true

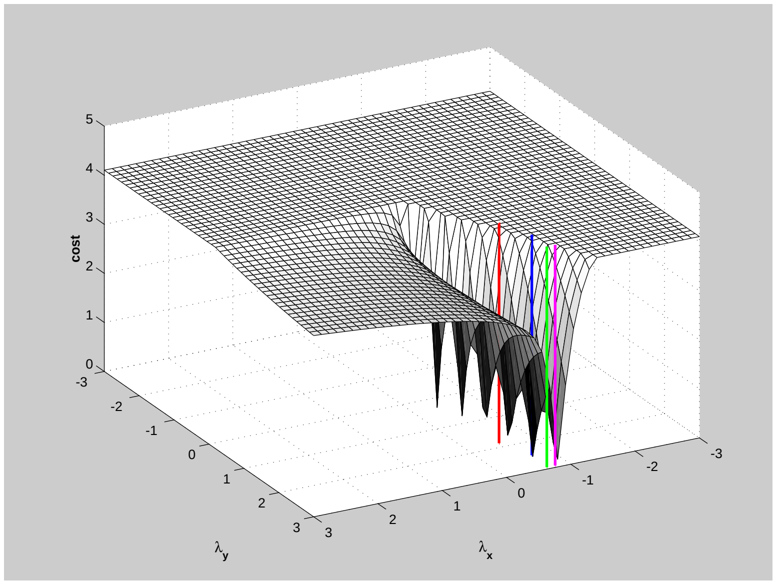


Figure 5. Cost as a function of costates

global minimum, the green line the downhill simplex solution, the blue line the simulated annealing solution and the magenta line the genetic algorithm solution. Considering the apparent simplicity of the problem the cost plot is surprisingly complex. A large part of the plot is a plateau. A gradient based method is likely to fail if the initial guess is in this region. The region in the lower left hand corner slopes towards the canyon. However, the canyon has many local minima. Thus even if a solution falls into the canyon it is possible that it will fall into a local minima. On the other hand, it may not be critical that the solution reach the true global minima since the other minima have low cost and may be adequate solutions to the problem.

VII. Trajectory Equations

A. Coordinate Systems

The choice of coordinates depends on the boundary conditions. It is desirable to have the boundary conditions be fixed values of the states. For example, if the problem is a minimum time transfer to a particular point in space then cartesian coordinates will work well. If the goal is to achieve a set of orbital elements then Keplerian or equinoctial equations are best suited for the problem. Although the elements can be found as functions of cartesian velocity and position, the derivatives of those functions have discontinuities making it difficult to have inclination or eccentricity as boundary conditions. Equinoctial elements have the advantage of not having a singularity when the inclination or eccentricity is zero which is the case for many orbits of interest. Although the equinoctial elements do not include inclination and eccentricity explicitly both are simple functions of the equinoctial elements.

In this paper we will use cylindrical coordinates for the 2 dimensional problem and equinoctial elements for the 3 dimensional problem.

B. Derivation of the Two Dimensional Equations

The central body equations for a spacecraft orbiting the sun are

$$\ddot{x} + \mu \frac{x}{R^3} = a \tag{22}$$

where x is the cartesian position vector, μ is the gravitational constant of the sun, R is the distance from the central body and a is the acceleration from the thruster.

This equation assumes that the spacecraft is acted upon only by the central force, in this case the sun. Accelerations due to other sources such as radiation pressure, the gravitational acceleration due to other solar system bodies, higher order harmonics of the central body, are neglected. Were the spacecraft in planetary orbit it would be necessary to include aerodynamic drag as an external force. It should be noted that for interplanetary optimization the addition of planetary perturbations adds a considerable amount of computation due to the complexity of the ephemerides of the planets.

Define

$$\begin{aligned} x &= r\cos\theta \\ y &= r\sin\theta \end{aligned} \tag{23}$$

where θ is measured from the x-axis. z is unchanged between cylindrical and cartesian coordinates.

Taking derivatives of x and y

$$\begin{aligned} \dot{x} &= \dot{r}\cos\theta - r\dot{\theta}\sin\theta \\ \dot{y} &= \dot{r}\sin\theta + r\dot{\theta}\cos\theta \end{aligned} \tag{24}$$

$$\begin{aligned} \ddot{x} &= \ddot{r}\cos\theta - 2\dot{r}\dot{\theta}\sin\theta - r\ddot{\theta}\sin\theta - r\dot{\theta}^2\cos\theta \\ \ddot{y} &= \ddot{r}\sin\theta + 2\dot{r}\dot{\theta}\cos\theta + r\ddot{\theta}\cos\theta - r\dot{\theta}^2\sin\theta \end{aligned} \tag{25}$$

The equations in cylindrical coordinates are

$$\ddot{r} - r\dot{\theta}^2 + \frac{\mu}{R^2} = a_x\cos\theta + a_y\sin\theta \tag{26}$$

$$r\ddot{\theta} + 2\dot{r}\dot{\theta} = a_y\cos\theta - a_x\sin\theta \tag{27}$$

$$\ddot{z} + \frac{\mu z}{R^3} = a_z$$

where a is the disturbance acceleration vector in cartesian coordinates and

$$R = \sqrt{r^2 + z^2} \tag{28}$$

We add the equation

$$\dot{m} = -T/u_e \tag{29}$$

where T is the thrust and u_e is the exhaust velocity. The in-plane accelerations are replaced by the radial and tangential components of the acceleration.

$$\begin{aligned} a_r &= a_x \cos\theta + a_y \sin\theta \\ a_\theta &= a_y \cos\theta - a_x \sin\theta \end{aligned} \tag{30}$$

The acceleration in the 2D equations is for a constant thrust.

$$a = \frac{T}{m} \tag{31}$$

The state equations are now

$$f = \begin{bmatrix} u \\ \frac{v^2}{r} - \frac{\mu}{r^2} + \frac{T}{m}\sin\phi \\ -\frac{uv}{r} + \frac{T}{m}\cos\phi \\ -\frac{T}{u_e} \end{bmatrix} \tag{32}$$

The partials of the state equations become

$$f_x = \begin{bmatrix} 0 & 1 & 0 & 0 \\ -\frac{v^2}{r^2} + \frac{2\mu}{r^3} & 0 & \frac{2v}{r} & -\frac{T\sin\phi}{m^2} \\ \frac{uv}{r^2} & -\frac{v}{r} & -\frac{u}{r} & -\frac{T\cos\phi}{m^2} \\ 0 & 0 & 0 & 0 \end{bmatrix} \tag{33}$$

The optimality condition becomes

$$\frac{\lambda_u}{\lambda_v} = \tan\phi \tag{34}$$

This defines the optimal thruster control angle ϕ for the thruster. For the initial guess of the λ vector we compute

$$\lambda_u = \lambda_v \tan\phi \tag{35}$$
$$\lambda_r = 0 \tag{36}$$

For very low-thrust transfers the thrust vector should be tangential to the orbit, i.e. $\phi = 0$ so $\lambda_u = 0$ and λ_v is any value. The boundary conditions for the state equations are

$$x_0 = \begin{bmatrix} r_0 \\ 0 \\ \sqrt{\frac{\mu}{r_0}} \end{bmatrix} \tag{37}$$

$$x_f = \begin{bmatrix} r_F \\ 0 \\ \sqrt{\frac{\mu}{r_f}} \end{bmatrix} \tag{38}$$

C. The Three Dimensional Equations of Motion

The three dimensional equations of motion employ modified equinoctial coordinates. The modified coordinates are

$$\begin{aligned}
p &= a(1-e^2) && (39)\\
f &= e\cos(\omega+\Omega) && (40)\\
g &= e\sin(\omega+\Omega) &&\\
h &= \tan(i/2)\cos\Omega &&\\
k &= \tan(i/2)\sin\Omega &&\\
L &= \Omega+\omega+\nu &&
\end{aligned}$$

where p is the semiparameter, a is the semimajor axis, e is the orbital eccentricity, ω is the argument of perigee, Ω is the right ascension of the ascending node, L is the true longitude and ν is the true anomaly.

The dimensional dynamical equations are

$$\dot{x} = Gu + b \tag{41}$$

where the state vector is

$$x = \begin{bmatrix} p \\ f \\ g \\ h \\ k \\ L \\ m \end{bmatrix} \tag{42}$$

where m is the mass

$$G = \begin{bmatrix} 0 & 2r\omega & 0 \\ \omega s & \omega(\gamma c + f)/q & -\Omega g \\ -\omega c & \omega(\gamma s + g)/q & -\Omega f \\ 0 & 0 & \zeta c \\ 0 & 0 & \zeta s \\ 0 & 0 & \Omega \\ 0 & 0 & 0 \end{bmatrix} \tag{43}$$

and

$$b = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ \sqrt{\frac{\mu p}{r^2}} \\ -\frac{T}{u_e} \end{bmatrix} \tag{44}$$

where

$$\begin{aligned}
r &= \frac{p}{q} && (45)\\
c &= \cos L && (46)\\
s &= \sin L && (47)\\
z &= hs - kc && (48)\\
\omega &= \sqrt{\frac{p}{\mu}} && (49)
\end{aligned}$$

$$\begin{aligned} q &= 1 + fc + gs & (50)\\ \gamma &= q + 1 & (51)\\ \Omega &= \omega\frac{z}{q} & (52)\\ \zeta &= \frac{1}{2}\frac{\omega}{q}(1 + \sqrt{h^2 + k^2}) & (53) \end{aligned}$$

a is the acceleration vector with components in the radial, tangential and normal directions. If the accelerations are zero, only L changes. The last equation is for the mass flow of the thruster.

If we define the control acceleration vector as

$$u = a \begin{bmatrix} \cos\alpha\cos\beta \\ \sin\alpha\cos\beta \\ \sin\beta \end{bmatrix} \tag{54}$$

then

$$U = \begin{bmatrix} -\sin\alpha\cos\beta & -\cos\alpha\sin\beta \\ \cos\alpha\cos\beta & -\sin\alpha\sin\beta \\ 0 & \cos\beta \end{bmatrix} \tag{55}$$

Then the optimality condition becomes

$$0 = U^T G^T \lambda \tag{56}$$

which is two equations in two unknowns, α and β. This reduces to

$$\begin{aligned} 0 &= u_{11}\epsilon_1 + u_{21}\epsilon_2 & (57)\\ 0 &= u_{12}\epsilon_1 + u_{22}\epsilon_2 + u_{32}\epsilon_3 & (58) \end{aligned}$$

where $\epsilon = G^T\lambda$. Solving for the angles

$$\begin{aligned} \tan\alpha &= \frac{\epsilon_1}{\epsilon_2} & (59)\\ \tan\beta &= \frac{\epsilon_3}{\epsilon_1\cos\alpha + \epsilon_2\sin\alpha} & (60) \end{aligned}$$

It is desirable to be able to obtain a good initial guess for the costates when using the downhill simplex or simulated annealing methods. This can be derived if we can guess the initial thrust vector which then gives the initial α and β. The costate for the mass equation λ_m drops out but this still leaves two equations in 6 unknown costates. This means that we could set 4 parameters and solve for the remaining two and get the correct thrust direction. Another possibility is to define a cost and find initial parameters that achieve the correct thrust direction and minimize the cost. However, their is no guarantee that this would result in an initial costate vector that was close to the desired solution.

VIII. 2D Results

The two dimensional problem is for a transfer between a circular Earth orbit and a circular Mars orbit. Thrust is constant and the only control is the thrust angle in the orbit plane. Since the boundary conditions can be written explicitly in terms of the states, we can solve the problem by guessing the initial costates and propagating until some end time. The selection of the end time is problematic. If it is too short, i.e. less than the minimum, the optimizer will never solve the problem. If it is too long, the optimizer will be slow.

The second issue is when to stop the integration. Two approaches were investigated. One was to stop when the propagated trajectory reached the Mars orbit radius. ode113 will iterate on an end condition so this is achieved very accurately. This, in effect, constrains on of the boundary condition errors to be zero. In addition, it precludes any solution that crosses the Mars orbit and then returns. The second method is to propagate to a fixed end time and then pick the point where the trajectory has the smallest error.

There are four costates for this problem

$$\lambda = \begin{bmatrix} \lambda_r \\ \lambda_u \\ \lambda_v \\ \lambda_m \end{bmatrix} \tag{61}$$

However the four costates are related by the Hamiltonian

$$0 = \lambda_r f_r + \lambda_u f_u + \lambda_v f_v + \lambda_m f_m + 1 \tag{62}$$

f_m is always nonzero in a constant thrust problem therefore

$$\lambda_m = -\frac{1 + \lambda_r f_r + \lambda_u f_u + \lambda_v f_v}{f_m} \tag{63}$$

Thus the search algorithm only needs to search on three costates. This equation is identical in form to the transversality condition that would exist if H were an explicit function of t only this applies to the entire trajectory, not just the end.

The eight boundary conditions are

$$\begin{aligned} x(0) &= x_0 && (64)\\ r(t_f) &= \frac{r_{Earth}}{r_{Mars}} && (65)\\ u(t_f) &= 0 && (66)\\ v(t_f) &= \sqrt{\frac{r_{Mars}}{r_{Earth}}} && (67)\\ \lambda_m(t_f) &= 0 && (68) \end{aligned}$$

x is a vector of the four states. t_f and the four initial costates are part of the solution.

The thrust for the problem is 2 N and the exhaust velocity is 29.5 km/s. The mass of the spacecraft is initially 4536 kg and the fraction of that mass that is fuel is not specified.

The weights used for the four errors in these examples were

$$w = \begin{bmatrix} 1 & 3 \times 10^6 & 1 \times 10^7 & 1 \times 10^{-2} \end{bmatrix} \tag{69}$$

This roughly equalizes the numerical values of the states. The errors chosen for the 2D problem were the states themselves. This selection is not without problems, however. The desired radial rate is zero but in this example the initial radial rate is also zero. Therefore, if this value is weighted too heavily the algorithm will may stay in a local optima which is the initial orbit. Thus selecton of weights for the optimizer is not a simple task. One could base the weights on the equivalent velocity change to correct an error in that element, but this would add considerable complexity to the process.

Table 5 gives the errors in each case.

Table 5: 2D Solutions

Error	Downhill Simplex	Simulated Annealing	GAOT
Radial (km)	69,966	16,119,202	188,061
Radial Velocity (km/s)	-0.0187	9.67	0.0523
Tangential Velocity (km/s)	-0.0041	-0.843	0.0164
Time (days)	251	168	168
λ_r	0.0296	0.00359	0.0248
λ_u	45489	-99960	38126
λ_v	188917	100015	158223

The 2D trajectory plot and orbital elements are shown in Figure 6. An issue that arises with this approach is that it is very sensitive to the resolution of the numerical integration. Changing the resolution can cause the optimizer to not find any reasonable solutions. This can be solved by forcing the integrator to use small steps, but this defeats the purpose of a variable order, variable step size integrator and can increase the time required for a solution.

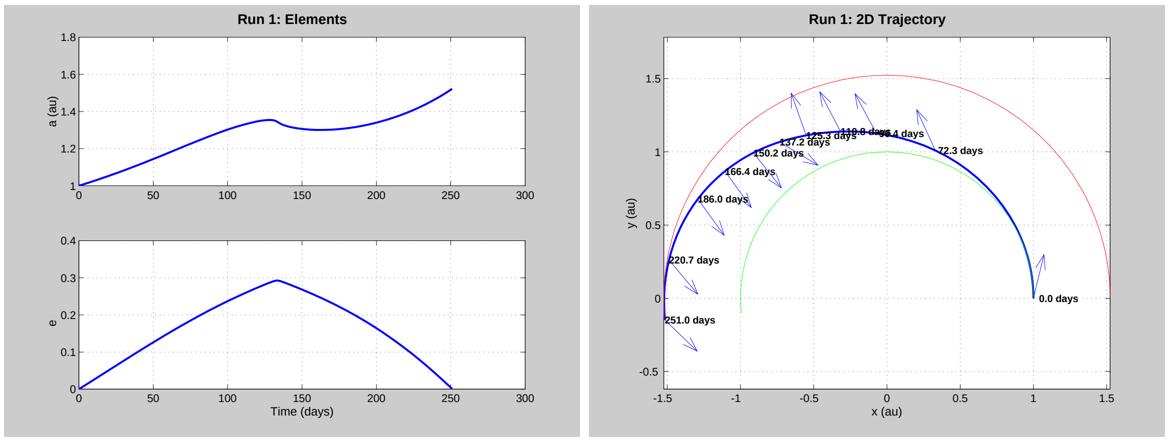


Figure 6. Typical 2D Mars trajectory

GAOT and downhill simplex produced comparable results but simulated annealing was unable to find a solution. Its estimate of the costates did not move significantly from the initial guess of

$$\lambda = \begin{bmatrix} 0 & -100000 & 100000 \end{bmatrix} \tag{70}$$

IX. 3D Results

The three dimensional problem is for a transfer to a 10 degree circular heliocentric orbit with a right ascension of the ascending node aligned with the x-axis. The initial and final radii are the same. The boundary conditions for the 3 dimensional problem are

$$x(t_0) = x_0 \tag{71}$$

At t_f we want

$$\begin{aligned} i(t_f) &= i_f & (72)\\ \Omega(t_f) &= \Omega_f & (73)\\ e(t_f) &= 0 & (74)\\ a(t_f) &= a_f & (75) \end{aligned}$$

where i_f is the final orbit inclination, Ω_f is the final right ascension of the ascending node and a_f is the final semi-major axis. Because $e = 0$ the argument of perigee ω is undefined. We don't care about the mean anomaly. Converting into equinoctial coordinates we find

$$\begin{aligned} p_f &= a_f & (76)\\ f_f &= g_f = 0 & (77)\\ h_f^2 + k_f^2 &= \tan^2 \frac{i_f}{2} & (78)\\ \frac{k_f}{h_f} &= \tan \Omega_f & (79) \end{aligned}$$

Solving for h

$$h_f^2 (1 + \tan^2 \Omega) = \tan^2 \frac{i_f}{2} \tag{80}$$

or

$$h_f = \frac{\tan\frac{i_f}{2}}{\sqrt{1+\tan^2\Omega}} \tag{81}$$

and

$$k_f = \frac{\tan\Omega\tan\frac{i_f}{2}}{\sqrt{1+\tan^2\Omega}} \tag{82}$$

Since we don't care about L then

$$\lambda_L(t_f) = 0 \tag{83}$$

Similar to the two dimensional problem,

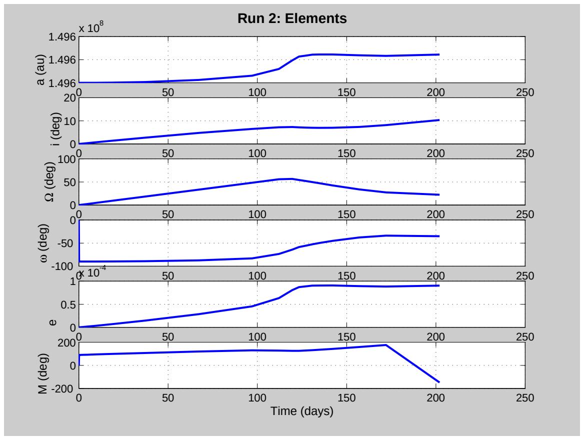
$$\lambda_m = -\frac{1+\lambda_p f_p + \lambda_f f_f + \lambda_g f_g + \lambda_h f_h + \lambda_k f_k + \lambda_L f_L}{f_m} \tag{84}$$

Formulation of the scalar cost is a function of the type of problem to be solved. Often the final and initial orbits are defined in terms of orbital elements. However, in many cases, some of the orbital elements will not change from the initial to the final orbit. For example, in a transfer between two coplanar circular orbits both the initial and final eccentricities and inclinations are zero. Thus the initial orbit is a local minima. The eccentricity problem can be solved by using apogee and perigee radii where

$$r_p = a(1-e) \tag{85}$$
$$r_a = a(1+e) \tag{86}$$

instead of semi major axis and eccentricity but this fix would not help if it was desired to just change the apogee radius and hold perigee radius fixed. In that case using semi-major axis and eccentricity in the cost function would be a better idea.

The 3D example is a transfer from a zero degree inclined orbit at the Earth's radius to a 10 degree inclined orbit. The 3D trajectory plot and orbital elements are shown in Figure 7 from the downhill simplex method.



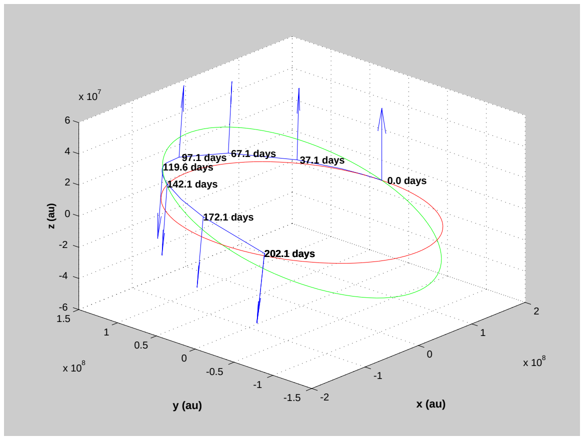


Figure 7. Inclination change maneuver

The eccentricity and semi-major axis errors remain small. The inclination reaches its proper value and the right ascension of the ascending node is approaching the correct value.

The elements obtained by GAOT are shown in Figure 8. The optimization gets the inclination and right ascension of the ascending node correct at the expense of an eccentricity and semi-major axis error.

X. Conclusions

With a reasonable amount of work by an analyst, the global methods employed in this paper can be used to generate optimal trajectories for low-thrust spacecraft. The algorithms demonstrated are readily available to the reader. Future work will explore alternative versions of these algorithms and other global

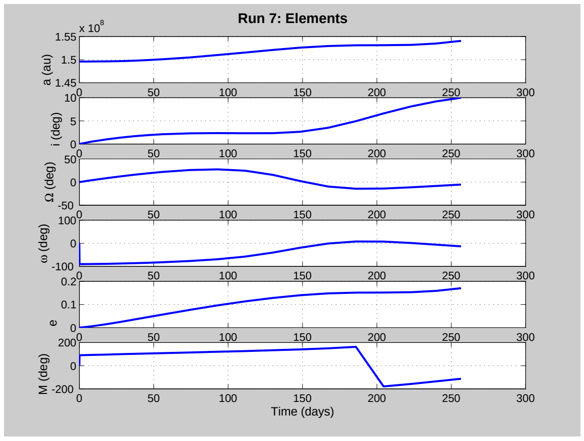


Figure 8. Inclination change maneuver using GAOT

optimization algorithms. Numerical issues, for example the use of normalized versus non-normalized states will be studied. The effects of different formulations for the cost function will be investigated. Finally, the issue of determining reasonable first guesses for the costates in the equinoctial case must be addressed when using simulated annealing or downhill simplex.

Acknowledgments

The authors would like to thank Professor Yiyuan Zhao for his helpful comments on this paper.

References

[1]Bryson, A. E. and Ho, Y., *Applied Optimal Control*, Hemisphere Publishing Company, 1975.

[2]Edelbaum, T. N., Sackett, L. L., and Malchow, H. L., "Optimal Low Thrust Geocentric Transfer," No. 73-1074, Oct 1973.

[3]Gath, P. F., *CAMTOS - A Software Suite Combining Direct and Indirect Trajectory Optimization Methods*, Ph.D. thesis, Institut fur Flugmechanik und Flugregelung Universitat Stuttgart, Nov 2002.

[4]Kim, M., *Continuous Low-Thrust Trajectory Optimization: Techniques and Applications*, Ph.D. thesis, Virginia Polytechnic Institute and State University, April 2005.

[5]Whiting, J. K., *Orbital Transfer Trajectory Optimization*, Master's thesis, Massachusetts Institute of Technology, Feb 2004.

[6]T.Betts, J., "Survey of Numerical Methods for Trajectory Optimization," *Journal of Guidance, Control, and Dynamics*, Vol. 21, No. 2, March-April 1998, pp. 193–207.

[7]Chernov, A. V., "OPTIMAL FLIGHT TO NEAR-EARTH ASTEROIDS WITH USING ELECTRIC PROPULSION AND GRAVITY MANEUVERS," .

[8]Cano, J. L., Bello-Mora, M., and Rodriguez-Canabal, J., "NAVIGATION AND GUIDANCE FOR LOW-THRUST TRAJECTORIES, LOTNAV," .

[9]Betts, J. T., *Practical Methods for Optimal Control Using Nonlinear Programming*, SIAM, 2001.

[10]Dachwald, B., *Low-Thrust Trajectory Optimization and Interplanetary Mission Analysis Using Evolutionary Neurocontroller*, Ph.D. thesis, Universitat der Bundeswehr Munchen, April 2004.

[11]Kim, M. and Hall, C. D., "Symmetries in the Optimal Control of Solar Sail Spacecraft," .

[12]P. J. Gage, R. D. Braun, I. M. K., "Interplanetary Trajectory Optimization Using a Genetic Algorithm," No. AIAA-94-3773-CP, Aug 1994.

[13]Nelder, J. A. and R., M., "Downhill Simplex," *Computer Journal*, Vol. 7, No. 2, 1965, pp. 308–313.

[14]Sipper, M., "A Brief Introduction to Genetic Algorithsm," *http://www.cs.bgu.ac.il/ sipper/ga.html*.

[15]Goffe, W., "Simulated Annealing - Global Optimization Method That Distinguishes Between Different Local Optima," *http://emlab.berkeley.edu/Software/abstracts/goffe895.html*.

[16]Press, W. H., Flannery, B. P., Teukolsky, S. A., and Vetterling, W. T., *Numerical Recipes*, Cambridge University Press, 1980.

[17]MathWorks, *Matlab*, MathWorks, 2005.

[18]GAOT, "GAOT," *http://www.*