JOURNAL OF GUIDANCE, CONTROL, AND DYNAMICS
Vol. 38, No. 11, November 2015

Hybrid Algorithm for Multiple Gravity-Assist and Impulsive Delta-V Maneuvers

Sam Wagner* and Bong Wie†
Iowa State University, Ames, Iowa 50011

DOI: 10.2514/1.G000874

A combination of multiple gravity-assist and impulsive Delta-V maneuvers is often required for interplanetary and interstellar space missions, such as NASA's Voyager 1 and 2, Galileo, and Cassini missions. The design of such complex interplanetary missions is difficult with traditional mission analysis techniques because the mission must be prepruned to determine potential trajectories. Prepruning has been necessary because these mission often require the optimization of dozens of variables in a highly nonlinear and discontinuous design space. This process risks pruning nonintuitive solutions, which may potentially contain the optimal trajectory. In this paper, a hybrid optimization algorithm that is capable of determining optimal interplanetary trajectories, including the number of gravity assists and the planetary flyby order, is developed. The hybrid optimization algorithm uses a stochastic genetic algorithm to globally search the design space, as well as traditional nonlinear programming gradient-based optimization tools to determine locally optimal trajectories. By combining the global convergence properties of genetic algorithms with the accuracy of gradient-based solvers, a robust optimization algorithm capable of determining near-optimal solutions for a complex interplanetary space mission is developed.

Nomenclature

a	=	orbit semimajor axis
E	=	orbit energy
e	=	gravity-assist eccentricity
N_{sc}	=	spacecraft resonant orbit number
$N_{\oplus}$	=	planet resonant orbit number
R_{pi}	=	planetary flyby radius
$R_{\oplus}$	=	planet radius
$\boldsymbol{r}$	=	radius vector
r_p	=	gravity-assist perigee radius
r_{pi}	=	radius ratio
$T_{\text{ecl}}^{\text{VNC}}$	=	transformation matrix for velocity-normal-conormal frame to ecliptic frame
$\boldsymbol{V}$	=	various velocity vectors in the heliocentric frame
$\boldsymbol{v}_\infty$	=	spacecraft incoming and outgoing velocity vectors with respect to the planet
α	=	ascension angle
β	=	declination angle
γ	=	B-plane insertion angle
Δt	=	specified time of flight
ΔV_{GA}	=	variable number
δ	=	gravity-assist turning angle
$\mu_{\oplus}$	=	planetary gravitational parameter

I. Introduction

SINCE the launch of Sputnik 1, on 4 October 1957, the human race has been driven by curiosity to explore beyond Earth and enter into the solar system. NASA's Mariner 2 spacecraft, launched on 27 August 1962, was the first successful interplanetary space probe, passing within 22,000 miles of Venus. Since then, a myriad of spacecraft have been launched in an effort to explore our solar system, although only a fraction of our solar system has been explored. Larger spacecraft and scientific payloads will be required to further explore the solar system. These missions can only be enabled through increasingly complex trajectories, often by the use of a combination of planetary gravity assists, deep-space correction maneuvers, and more recently, low-thrust solar electric propulsion.

To understand the significance of gravity assists and their importance to interplanetary missions, it is important to understand that the classical theories of space travel, as developed by notable rocket pioneers such as Hohmann [1], Goddard [2], and Tsiolkovskii and Petroff [3]. Before the invention of gravity assists, all spacecraft relied entirely on chemical rocket propulsion. This means there was a high-energy barrier, which prohibited exploration mission beyond the Earth's moon, Mars, and Venus. In essence, the ΔV necessary for exploring the solar system beyond our nearest neighbors is simply larger than what is feasible from traditional chemical rocket propulsion.

By the 1950s, initial work in advancing interplanetary trajectories was being performed by many notable orbital mechanics specialists such as Crocco [4], Ruppe [5], Battin [6], Lawden [7], Breakwell and Gillespie [8], and many others. By 1959, Battin [6] developed a method for computing round-trip free-return trajectories using solutions to Lambert's problem but had stopped short of calculating the trajectories because a mathematical method to represent the three-dimensional flyby trajectory had not yet been developed. In 1961, Minovitch [9] developed a new method to represent three-dimensional conic orbits with two orbital vectors, commonly known as $\boldsymbol{e}$ and $\boldsymbol{h}$. With this new method, he was able to develop what is now known as the patched conic method and began to calculate trajectories using gravity assists. Trajectories using gravity assists are able to obtain higher orbital energies than previously possible using only chemical rocket propulsion, opening up new opportunities for further exploration of our solar system.

To this day, gravity assists are commonly used for interplanetary and interstellar missions. A few notable missions that have used gravity assists include Mariner 10, Pioneer 10, Pioneer 11, Voyager 1, Voyager 2, Galileo, Cassini, Rosetta, and Messenger. The Voyager and Pioneer missions used gravity assists to insert the spacecraft on solar system escape trajectories, whereas Mariner 10, Galileo, Cassini, and Messenger missions exploited inner-planetary gravity assists to reach Mercury, Jupiter, and Saturn, respectively, without the need for heavy-lift launch vehicles [10–12]. As missions become increasingly complex, determining feasible trajectories quickly becomes a daunting task.

These types of interplanetary and interstellar mission design problems present significant optimization challenges. They are extremely nonlinear, often with multiple strong basins of attraction in the

Received 9 July 2014; revision received 29 April 2015; accepted for publication 3 May 2015; published online 10 August 2015. Copyright © 2015 by Samuel Wagner. Published by the American Institute of Aeronautics and Astronautics, Inc., with permission. Copies of this paper may be made for personal or internal use, on condition that the copier pay the $10.00 per-copy fee to the Copyright Clearance Center, Inc., 222 Rosewood Drive, Danvers, MA 01923; include the code 1533-3884/15 and $10.00 in correspondence with the CCC.

*Graduate Research Assistant, Department of Aerospace Engineering, Asteroid Deflection Research Center.

†Vance Coffman Endowed Chair Professor, Department of Aerospace Engineering, Asteroid Deflection Research Center.

neighborhood of optimal solutions, and are discontinuous when the planetary gravity-assist order is varied. The traditional method for solving complex impulsive mission design problems is to have the designer prune the decision space in order to obtain acceptable solutions. However, without robust automated methods to optimize such trajectories, globally optimal solutions may be overlooked by even the most experienced mission designer. Several algorithms have been proposed to automate the optimization for these types of missions. These algorithms are typically broken into two types: two-level and one-level algorithms. Two-level algorithms, commonly known as the hybrid optimal control problem [13–15], typically break the problem down into two subproblems. At the first level of the two-level algorithms, the planetary flyby sequence is optimized using an integer optimization method. At the second level, each trajectory is optimized for each flyby sequence from the first level. The one-level approach combines both the flyby sequence order and trajectory optimization in a mixed integer optimization problem [16–18]. Using the proposed hybrid optimization one-level algorithm, it will be shown that, when used properly, this new algorithm can find optimal or near-optimal solutions in a more efficient manner (shorter runtimes) than other current two-level multiple gravity-assist (MGA) and multiple gravity-assist with deep-space maneuver (MGA-DSM) optimization algorithms.

The proposed hybrid optimization algorithm combines stochastic evolutionary optimization techniques with traditional gradient-based nonlinear programming (NLP) methods in an effort to develop an optimization algorithm that can determine near-globally optimal interplanetary trajectories. The new algorithm is capable of finding near-optimal solutions for complex missions with multiple flybys and impulsive ΔV maneuvers, similar to NASA's Galileo and Cassini missions [19,20]. It is also capable of optimizing the number of gravity assists and planetary flyby order using a concept known as "hidden genes" [17]. The results will be compared with the past Galileo and Cassini missions in an effort to validate both the problem formulation and the hybrid optimization algorithm itself. The ultimate goal is to develop mission design tools that can efficiently and autonomously determine near-optimal trajectories for complex mission architectures.

Two classes of problems are developed and tested. The first are traditional multiple powered gravity-assist (MGA) missions, such as the Voyager or Galileo missions. The second mission type uses multiple gravity assists with the option for a deep-space maneuver during each interplanetary leg of the mission. This mission type is often referred to as the MGA-DSM model and was required for the Cassini mission [12–15,21,22].

II. Problem Formulation

Depending on mission requirements, trajectories can be modeled by either two-impulse or three-impulse trajectories. Missions that do not require large deep-space maneuvers can be modeled with the two-impulse model, commonly known as the multiple gravity-assist model. Missions such as Mariner 10, Pioneer 10/11, Voyager 1/2, and Galileo can all be modeled with the MGA model. These types of missions require far fewer variables than the three-impulse MGA-DSM model, allowing significantly decreased computational costs when compared to missions requiring three-impulse transfers.

The three-impulse, or MGA-DSM, model is an extension of the simple two-impulse model [16,23]. This model simply propagates the orbit by a given amount of time before performing a deep-space maneuver to target the new planet in the trajectory sequence. The purpose of outlining both of these methods is to formulate a way to determine objective functions for the optimization algorithm. The objective functions typically consists of required mission ΔV, the Earth departure V_∞, the final arrival V_∞, and any other mission constraints. Any permutation of these elements can be used to meet specific requirements for each mission.

A. Multiple Gravity-Assist Model

With the multiple gravity-assist two-impulse model, planetary flybys are modeled using the three-dimensional patched conic method developed by Minovitch [9]. The flyby orbit geometry is defined by the incoming and outgoing spacecraft velocity vectors, which are given by two solutions to Lambert's problem [24]. This patching results in a powered hyperbolic orbit for each gravity assist that requires a ΔV at the perigee of each flyby. It should be noted that the ΔV required for each gravity assist is typically driven to a near-zero value, resulting in a free flyby.

For each gravity assist, the incoming and outgoing velocity vectors (in the heliocentric frame) are given directly from solutions to Lambert's problem [25–29]. The incoming and outgoing velocity vectors, relative to each planet, are then found by subtracting the planet's velocity from the incoming and outgoing spacecraft (s/c) velocities, given from the solutions to Lambert's problems as follows:

$$\boldsymbol{v}_{\infty-\text{in}} = \boldsymbol{V}_{\text{s/c}-\text{in}} - \boldsymbol{V}_{\oplus} \tag{1}$$

$$\boldsymbol{v}_{\infty-\text{out}} = \boldsymbol{V}_{\text{s/c}-\text{out}} - \boldsymbol{V}_{\oplus} \tag{2}$$

The perigee radius, which is required to patch the two solutions together, is a function of the incoming and outgoing orbits. The first step is to determine the semimajor axis of the incoming and outgoing hyperbolic trajectories as

$$a_{\text{in}} = -\frac{\mu_\oplus}{v^2_{\infty-\text{in}}} \tag{3}$$

$$a_{\text{out}} = -\frac{\mu_\oplus}{v^2_{\infty-\text{out}}} \tag{4}$$

where $\mu_\oplus$ is the target planet's gravitational parameter.

The required turning angle for the gravity assist is a function of the incoming and outgoing v_∞ vectors, defined as

$$\delta = \cos^{-1}\left(\frac{\boldsymbol{v}_{\infty-\text{in}} \cdot \boldsymbol{v}_{\infty-\text{out}}}{v_{\infty-\text{in}} \cdot v_{\infty-\text{out}}}\right) \tag{5}$$

With this model the flyby perigee radius must be equal for both legs of the hyperbolic orbit, as described by

$$r_p = a_{\text{in}}(1 - e_{\text{in}}) = a_{\text{out}}(1 - e_{\text{out}}) \tag{6}$$

where e_{in} and e_{out} are the incoming and outgoing orbit eccentricities. It should be noted that two eccentricities should be greater than one because hyperbolic incoming and outgoing orbits are required to avoid being captured by the planet. The turning angle δ can also be represented as the sum of the transfer angles for the incoming and outgoing orbits, represented as follows:

$$\delta = \sin^{-1}\left(\frac{1}{e_{\text{in}}}\right) + \sin^{-1}\left(\frac{1}{e_{\text{out}}}\right) \tag{7}$$

Equations (6) and (7) can be written into a single equation that can be iteratively solved in order to determine the incoming and outgoing eccentricities as follows:

$$f = \left(\frac{a_{\text{out}}}{a_{\text{in}}}(e_{\text{out}} - 1)\right) \sin\left(\delta - \sin^{-1}\left(\frac{1}{e_{\text{out}}}\right)\right) - 1 = 0 \tag{8}$$

The preceding equation, which is a function of the unknown parameter e_{out}, must be solved with an iterative root-finding method. For this problem, a simple Newton root-finding method works well. To start the Newton iteration, an initial value for e_{out} of 1.5 works well. In a Newton iteration scheme, a while loop is used until e_{out} stops changing within a specified tolerance. The number of iterations should also be monitored, so the process can be terminated if a solution does not converge. The required first derivative of f with respect to e_{out} is

$$\frac{\mathrm{d}f}{\mathrm{d}e_{\text{out}}} = \left(\frac{a_{\text{out}}}{a_{\text{in}}} e_{\text{out}} - \frac{a_{\text{out}}}{a_{\text{in}}} + 1\right) \frac{\cos\left(\delta - \sin^{-1}(1/e_{\text{out}})\right)}{e_{\text{out}}^2 \sqrt{1 - \frac{1}{e_{\text{out}}^2}}} + \frac{a_{\text{out}}}{a_{\text{in}}} \sin\left(\delta - \sin^{-1}(1/e_{\text{out}})\right) \tag{9}$$

When a converged e_{out} is found, the perigee radius is calculated from Eq. (6). Finally, the ΔV required to patch the two orbits at the perigee can be determined as follows:

$$\Delta V_{\text{GA}} = \left| \sqrt{v_{\infty-\text{in}}^2 + \frac{2\mu_\oplus}{r_p}} - \sqrt{v_{\infty-\text{out}}^2 + \frac{2\mu_\oplus}{r_p}} \right| \tag{10}$$

The flyby perigee radius and ΔV are functions of the spacecraft's incoming and outgoing velocities. These are found from solutions to Lambert's problem, which are a function of planetary positions and time of flight (TOF). The planetary positions are also a function of time, so the only decision variables with this model are the date of Earth departure and time of flight for each additional leg of the trajectory. The final cost function C for the MGA optimization problem is represented as a function of the decision variables $\boldsymbol{X}$ as follows:

$$C = f(\boldsymbol{X}) + g(\boldsymbol{X}) \tag{11}$$

$$\boldsymbol{X} = [T_0, T_1 \ldots T_f]^T \tag{12}$$

where T_0 is the launch date, and T_i are the times of flight for each leg of the mission. The typical penalty function $g(\boldsymbol{X})$ will be discussed at the end of this section.

The final cost functions are then constructed by summing all of the required mission ΔV, as well as any other terms the user wishes to minimize, such as the initial departure or arrival v_∞. Globally optimal solutions of the MGA objective function(s) can then be found with the proposed hybrid optimization algorithm.

B. Multiple Gravity-Assist Deep-Space Maneuver Model

The MGA model, although simple and easy to implement, is not suitable for all missions. In many situations, allowing deep-space maneuver(s), where a ΔV is applied some time during the interplanetary coast arc, results in a significant reduction of the total required spacecraft ΔV. This implies that the optimal location for mission burns may not be at the flyby perigees. This section outlines the basics of the three-impulse MGA-DSM model [12,13,16,23,30].

The MGA-DSM model consists of three mission phases: Earth departure, gravity assist(s), and the final terminal phase. For the launch phase, an impulsive ΔV is applied at the Earth departure radius in a direction defined by the problem variables. The departure velocity vector is defined by three variables: the v_∞ magnitude (or, alternatively, C_3), and the departure right ascension and declination angles (α and β, respectively). Three additional variables are needed to completely define the departure phase; these are the launch date, time of flight from launch to the first flyby, and the burn index. The burn index ϵ defines a point along the trajectory in which an impulsive ΔV is applied to target the next planet in the trajectory. The burn index can range anywhere between zero and one, although both zero and one represent a simple Lambert solution without an additional correction maneuver. For the actual implementation, a maximum burn index of 0.99 and minimum burn index of 0.01 are used. This is done to ensure the Lambert solution algorithms are able to compute solutions. Directly specifying the departure declination has the added benefit of ensuring that all of the departure trajectories can be flown by a given launch vehicle. This is important because launch vehicles often have lowered C_3 performance when the departure declination is outside the launch vehicle's intended range. The spacecraft's initial state vector at the Earth departure is fully defined by four problem variables (the Earth departure date T_{launch}, v_∞, α, and β) as follows:

$$\boldsymbol{r}_{\text{s/c}} = \boldsymbol{r}_\oplus(T_0) \tag{13}$$

$$\boldsymbol{V}_{\text{s/c}} = \boldsymbol{V}_\oplus(T_{\text{launch}}) + v_\infty[\cos\alpha\cos\beta\,\boldsymbol{I} + \sin\alpha\cos\beta\,\boldsymbol{J} + \sin\beta\,\boldsymbol{K}] \tag{14}$$

From this point, the next step is essentially the same for both the departure phase and gravity-assist phases. The spacecraft's state vector is propagated forward by a time $\epsilon_1 T_1$, at which point a deep-space maneuver is applied to target the next planet. This targeting is performed by applying a solution to Lambert's problem, with the time of flight given by $(1 - \epsilon_1)T_l$. The ΔV required by the deep-space maneuver is the magnitude of the difference between the velocity vector after the orbit is propagated and the initial velocity vector from Lambert's solution. The spacecraft's planetary arrival velocity, also given from the solution to Lambert's problem, is then used for the next phase of the mission.

Each additional gravity assist in the trajectory requires an additional four variables. The required variables include the flyby radius ratio r_{pi}, the b-plane insertion angle γ_i, the time of flight T_i, and the burn index ϵ_i. The subscript i is an index specifying individual gravity assists. The radius ratio is multiplied by the flyby planet's radius to determine the flyby perigee radius as

$$R_{\text{pi}} = r_{\text{pi}} R_\oplus \tag{15}$$

Because the flyby is unpowered, the incoming and outgoing v_∞ magnitudes must be equal as

$$|\boldsymbol{v}_{\infty-\text{in}}| = |\boldsymbol{v}_{\infty-\text{out}}| \tag{16}$$

Similar to the MGA model, the incoming velocity vector is given by

$$\boldsymbol{v}_{\infty-\text{in}} = \boldsymbol{V}_{\text{s/c}-\text{in}} - \boldsymbol{V}_\oplus \tag{17}$$

The orientation of the outgoing velocity vector can be determined from the problem geometry. The flyby orbit's semimajor axis and eccentricity can be determined as follows:

$$a = -\frac{\mu_\oplus}{v_\infty^2} \tag{18}$$

$$e = 1 - \frac{R_{\text{pi}}}{a} \tag{19}$$

The flyby turning angle is a function of the flyby eccentricity, given by

$$\delta = 2 \arcsin \frac{1}{e} \tag{20}$$

The outgoing v_∞ direction is determined from the following equation:

$$\boldsymbol{v}_{\infty-\text{out}} = v_\infty[\cos\delta\hat{i} + \cos\gamma_i \sin\delta\hat{j} + \sin\gamma_i \sin\delta\hat{k}] \tag{21}$$

where $\hat{i}$, $\hat{j}$, and $\hat{k}$ are the b-plane unit vectors, defined as follows:

$$\hat{i} = \frac{\boldsymbol{v}_{\infty-\text{in}}}{|\boldsymbol{v}_{\infty-\text{in}}|} \tag{22}$$

$$\hat{j} = \frac{\hat{i} \times \boldsymbol{V}_{\infty-\text{in}}}{|\hat{i} \times \boldsymbol{V}_{\infty-\text{in}}|} \tag{23}$$

$$\hat{k} = \hat{i} \times \hat{j} \tag{24}$$

The final departure velocity is found by adding the current velocity of the planet with the outgoing v_∞ vector, and it is given by

$$\boldsymbol{V}_{\text{s/c-out}} = \boldsymbol{v}_{\infty\text{-out}} + \boldsymbol{V}_\oplus \tag{25}$$

As with the Earth departure phase, the spacecraft is propagated forward by $\epsilon_i T_i$ time. The next planet is then targeted with a solution to Lambert's problem, and the required deep-space maneuver is then calculated.

The terminal phase of the mission can be formulated in multiple ways. The deep-space maneuver that targets the final planet is part of the last flyby, so the only addition to the objective function is typically either the arrival v_∞ or a required ΔV for insertion into a specific orbit. In the case of the Cassini mission, the orbit about the final planet is defined by a specific insertion perigee radius and orbit eccentricity.

The final cost function is similar to the MGA model. However, the total number of state variables is significantly increased. As before, $f(\boldsymbol{X})$ is the object function (real mission ΔV, etc.) and $g(\boldsymbol{X})$ is a penalty function used to enforce problem constraints. In this case, n is the number of gravity assists required by the mission. The final cost function representation and the decision variables are defined by

$$C = f(\boldsymbol{X}) + g(\boldsymbol{X}) \tag{26}$$

$$\boldsymbol{X} = [T_0, v_{\infty-0}, \alpha_0, \beta_0, \epsilon_0, T_l, T_1 \ldots T_{n+1}, \epsilon_1 \ldots \epsilon_n, \gamma_1 \ldots \gamma_n, r_{p1} \ldots r_{\text{pn}}]^T \tag{27}$$

The objective functions are typically formulated to minimize the total mission ΔV or, alternatively, to maximize the final spacecraft arrival mass. An example of a typical objective function, without any constraint penalties, is chosen as

$$C = v_{\infty-0} + \Delta V_0 + \Delta V_1 \ldots \Delta V_n + v_{\infty-f} \tag{28}$$

where ΔV_i is the magnitude of the deep-space maneuvers corresponding a particular planetary flyby.

C. Problem Constraints

In optimization problems, constraints are often used to help shape the final solution and ensure only feasible solutions are found. When using optimization algorithms, constraints are used instead of hard variable limits. This ensures all possible solutions are continuous and can be optimized with the gradient-based hybrid optimization algorithm.

Common constraints for mission design problems include flyby altitude limits, mission time constraints, and penalties designed to protect against low-velocity flybys. During a low-velocity flyby, the spacecraft would be captured by the flyby planet rather than gaining velocity in the heliocentric frame. These are rare and can typically only occur at the first planet in the flyby sequence. Any additional constraint that helps shape the solution can be added, as long as the constraint values are approximately the same order of magnitude at the intended objective function values. It should be noted that, when using the MGA-DSM model in conjunction with the hybrid algorithm, all of the variables are explicitly constrained. However, low-velocity flybys may still occur.

The MGA model often results in a perigee radius that passes through the planet or close to the planet's atmosphere. In this situation, a constraint function to move the solution toward more feasible trajectories can be expressed as [13]

$$g_i(\boldsymbol{X}) = -2 \log \frac{R_{\text{pi}}}{kR_\oplus} \tag{29}$$

where k is a multiplier used to define how close the spacecraft is allowed to fly by a target planet. For the Cassini mission, a value of 1.05 is used. However, the Galileo mission had an Earth close approach altitude of 300 km corresponding to a k value of approximately 1.047.

The next constraint penalizes low-velocity flybys. This method has also been adapted from [13]. Low-velocity flybys are very rare, but solutions should still be protected from these unfeasible trajectories. The orbital energy, about the flyby planet, can be calculated as

$$E = \frac{|\boldsymbol{v}_{\infty-\text{in}}|^2}{2} - \frac{\mu_\oplus}{R_{\text{soi}}} \tag{30}$$

For the flyby orbit to be hyperbolic about the planet, E must be greater than zero. However, the sphere of influence model is an approximation, so an additional 10% margin on the incoming velocity is added. A simple penalty that is scaled inversely to the flyby velocity is used. For relatively large v_∞ values, the penalty is near zero and is small compared to the overall cost function value. Alternatively, for very low v_∞ values, the penalty is large enough to influence the final shape of the solution. The flyby orbital energies, adjusted for the 10% margin and final constraint, are calculated using the following two equations:

$$E = \frac{|0.9\boldsymbol{v}_{\infty-\text{in}}|^2}{2} - \frac{\mu_\oplus}{R_{\text{soi}}} \tag{31}$$

$$g_i(\boldsymbol{X}) = \begin{cases} 0 & E \geq 0 \\ \frac{1}{|\boldsymbol{v}_{\infty-\text{in}}|} & E < 0 \end{cases} \tag{32}$$

Additional penalty constraints can be added to shape the solution as desired. For MGA and MGA-DSM missions, the final constraint function is represented as the sum of each individual constraint by

$$g(\boldsymbol{X}) = \sum g_i(\boldsymbol{X}) \tag{33}$$

D. Constructing Resonant Orbit Flybys

Resonant orbits can occur when an MGA mission visits the same planet consecutively. If the required time of flight is such that the spacecraft returns to the same point for two flybys, then solutions to Lambert's problem are unable to determine a sufficient orbit. When this happens, the method outlined in this section is used to calculate the first gravity assist's outgoing velocity $\boldsymbol{v}_\infty$ and second gravity assist's incoming velocity, $\boldsymbol{v}_\infty$. The method outlined in this section is similar to [31].

If the transfer orbit period Δt is within ± 2 days of a resonant orbit, then the resonant orbit method outlined in this section is used to replace the solution to Lambert's problem for the arc between the two planetary flybys. A resonant orbit is tested for as follows:

$$\left| \Delta t - \frac{N_{\text{sc}}}{N_\oplus} T_\oplus \right| \leq 2\text{days} \tag{34}$$

where $T_\oplus$ is the orbital period of the flyby planet, and Δt is the required period for the transfer orbit between the two flybys. To be a resonant orbit, the transfer orbit time of flight must be an integer number of the planetary period $T_\oplus$. This is represented by the two integers N_{sc} and $N_\oplus$, where N_{sc} ranges from two to three and $N_\oplus$ ranges from one to two are checked. This allows for a wide range of resonant orbits to be checked: from 2:1 up to 3:2.

From the required transfer orbit time of flight Δt, the heliocentric semimajor axis is determined as

$$a = \left[\frac{\Delta t^2}{2\pi} \mu_{\text{sun}} \right]^{1/3} \tag{35}$$

where μ_{sun} is the gravitational parameter of the sun. Care should be taken to ensure all parameters are in the correct units.

From the semimajor axis, the magnitude of the heliocentric spacecraft velocity after the first gravity assist is calculated as follows:

$$|\boldsymbol{V}^{\text{GA1}}_{\text{sc-out}}| = \sqrt{\mu_{\text{sun}}\left(\frac{2}{|\boldsymbol{r}^{\text{GA1}}_{\oplus}|} - \frac{1}{a}\right)} \tag{36}$$

where the superscript GA1 indicates a parameter to describe the first flyby orbit, and $\boldsymbol{r}^{\text{GA1}}_{\oplus}$ is the planets radius at the time of the first flyby.

To compute the B-plane parameters of the first flyby two angles, θ and ϕ are used. The geometry of the velocity vectors after the first flyby is shown in Fig. 1. From the problem geometry, θ can be calculated from the law of cosines:

$$|\boldsymbol{V}^{\text{GA1}}_{\text{sc-out}}|^2 = |\boldsymbol{v}_{\infty}|^2 + |\boldsymbol{V}^{\text{GA1}}_{\oplus}|^2 - 2|\boldsymbol{v}^{\text{GA1}}_{\infty\text{-out}}||\boldsymbol{V}^{\text{GA1}}_{\oplus}| \cos\theta \tag{37}$$

This method requires unpowered gravity assists where

$$|\boldsymbol{v}_{\infty}| = |\boldsymbol{v}^{\text{GA1}}_{\infty\text{-in}}| = |\boldsymbol{v}^{\text{GA1}}_{\infty\text{-out}}| = |\boldsymbol{v}^{\text{GA2}}_{\infty\text{-in}}| \tag{38}$$

The variable ϕ is then used as the additional variable to define the flyby geometry.

Figure 1 shows there are infinitely many possible ϕ angles that would place the spacecraft into a resonant orbit. For this method, ϕ is then varied to determine possible transfer orbit geometries. The initial outgoing velocity vector, as a function of ϕ, is given as

$$\boldsymbol{v}^{\text{GA1}}_{\infty\text{-out}} = v_{\infty}[\cos(\pi-\theta)\hat{i} + \sin(\pi-\theta)\cos\phi\hat{j} - \sin(\pi-\theta)\sin\phi\hat{k}] \tag{39}$$

The outgoing velocity vector is expressed in a velocity-normal-conormal (VNC) reference frame, which varies with the orbit of the flyby planet. By transforming this vector from a VNC frame to the planetary ecliptic frame, the outgoing velocity is expressed in a reference frame that does not vary over time. The transformation matrix to transform the $\boldsymbol{v}_{\infty}$ vector to the planetary ecliptic frame is expressed as

$$T^{\text{VNC}}_{\text{ecl}} = [\hat{V}\,\hat{N}\,\hat{C}] \tag{40}$$

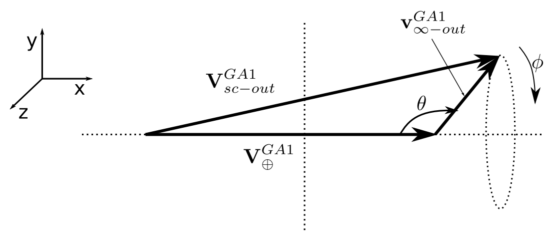
where $\hat{V}$, $\hat{N}$, and $\hat{C}$ are defined as

$$\hat{V} = \frac{\boldsymbol{V}_{\oplus}}{|\boldsymbol{V}_{\oplus}|} \tag{41}$$

$$\hat{N} = \frac{\boldsymbol{R}_{\oplus}\times\boldsymbol{V}_{\oplus}}{|\boldsymbol{R}_{\oplus}\times\boldsymbol{V}_{\oplus}|} \tag{42}$$

$$\hat{C} = \hat{V}\times\hat{N} \tag{43}$$

The incoming velocity for the second flyby, adjusted for slight variations in the planet's velocity, in the ecliptic frame is given as



y
x
z
$\mathrm{V}^{GA1}_{sc-out}$
$\mathrm{v}^{GA1}_{\infty-out}$
θ
ϕ
$\mathrm{V}^{GA1}_{\oplus}$

Fig. 1 Illustration of possible outgoing spacecraft orientations after the first gravity assist.

$$\boldsymbol{v}^{\text{GA2}}_{\infty\text{-in-ecl}} = \boldsymbol{v}^{\text{GA1}}_{\infty\text{-out-ecl}} + \boldsymbol{V}^{\text{GA1}}_{\oplus} - \boldsymbol{V}^{\text{GA2}}_{\oplus} \tag{44}$$

At this point, the incoming velocity is converted back, with the transformation matrix, to give the incoming velocity vector in the VNC frame. With $\boldsymbol{v}^{\text{GA1}}_{\infty\text{-out}}$ and $\boldsymbol{v}^{\text{GA2}}_{\infty\text{-in}}$ vectors now known, the algorithm computes the trajectory in the same way as the MGA trajectories.

The algorithm used to replace the solution to Lambert's problem for the resonant orbit flyby is given in the following. For this algorithm, one variable, ϕ, is required to define the trajectory for each resonant orbit. A maximum of two resonant orbits are allowed, resulting in two additional decision variables to be optimized. If a resonant orbit is not encountered, the two variables are treated as hidden genes, as described in the following section. Because the resonant orbit results in an unpowered flyby, there is no direct contribution to the objective function from the first resonant insertion gravity assists. The nonlinear perigee flyby radii constraints, as outlined in the previous section, are used to shape the solution and ensure the resulting trajectory does not pass too close to the planet during either close encounter.

1) Inputs are $\boldsymbol{v}^{\text{GA1}}_{\infty\text{-in}}$, $\boldsymbol{V}^{\text{GA1}}_{\oplus}$, $\boldsymbol{v}^{\text{GA2}}_{\infty\text{-out}}$, $\boldsymbol{V}^{\text{GA2}}_{\oplus}$, and ϕ.

2) Calculate the semimajor axis a and magnitude of the spacecraft velocity after the first flyby $\boldsymbol{V}^{\text{GA1}}_{\text{sc-out}}$ for the resonant orbit.

3) Compute θ and the outgoing velocity $\boldsymbol{v}^{\text{GA1}}_{\infty\text{-out}}$.

4) Use $T^{\text{VNC}}_{\text{ecl}}$ to determine the outgoing velocity in the ecliptic frame $\boldsymbol{v}^{\text{GA1}}_{\infty\text{-out-ecl}}$.

5) Compute the incoming velocity in the ecliptic frame $\boldsymbol{v}^{\text{GA2}}_{\infty\text{-in-ecl}}$ and transform it back to the VNC frame.

6) Use the incoming and outgoing $\boldsymbol{v}_{\infty}$ vectors to determine both flyby perigee radii as outlined for MGA missions.

7) From the two flyby radii, calculate the flyby constraints as previously outlined.

E. Problem Transcription Method

This section describes the method used to formulate the problem transcript, which is how each of the variables used that describe an individual solution is represented by the hybrid genetic algorithm (GA)-NLP solver. The method used to transcribe these problems is similar to those by Englander et al. [13–15,32], Gad and Abdelkhalik [17], and Abdelkhalik and Gad [18]. The problems are broken down into two variable sets: discrete integer variables and continuous real-valued variables. A single chromosome, or solution set, then consists of two sets of variables, the integer variables, and the real-valued variables. This approach is necessary because the NLP problem solver(s) are only able to iterate on real-valued continuous variables. However, the locally optimal solutions found by the NLP solver(s) do not affect the planetary flyby order or the number of gravity assists used for a single chromosome. The genetic algorithm is then used to explore the solution space of both the integer and real-valued variables. With this approach, previously developed NLP solvers are used in conjunction with the genetic algorithm implemented for this study [19,20].

A breakdown of each integer and the real-valued variables for the MGA missions are shown in Table 1. The integer portion of the chromosome for this mission type include the number of flybys that will be performed, a variable for each flyby planet, and the final arrival planet. The maximum number of flybys was allowed to vary between one and eight. An integer code was used to specify the flyby planets as follows: values of 1 to 8 specified the planet in order from Mercury to Neptune, 9 was Pluto, and 10 indicated that user-supplied ephemeris data will be used, typically for bodies such as comets or asteroids. The arrival planet was also coded as a variable so the user could specify the final arrival planet without hard coding it into the cost function. For the missions presented in this study, the upper and lower bounds for the arrival planet are set to the same value (ensuring the arrival planet is always the same). The real-value variables for the MGA missions include the departure date, time of flight for each gravity assist, the time of flight of the final leg to the arrival planet, and two additional ϕ angle variables, which are only used if resonant orbit(s) are encountered.

Table 1 Integer- and real-valued variables for the MGA problem

Integer	Real-valued variables
Number of gravity assists n	Departure date J_0
Gravity-assist planet: $P_1, \ldots, P_n$	Time of flight for each gravity assist: $T_1, \ldots, T_n$
Final arrival planet P_f	Time of flight for the final leg T_f
	Two ϕ angle variables for possible resonant orbits

For the MGA-DSM missions, the integer variables are set up the same as MGA missions. However, the number of real valued is significantly increased. The launch requires six variables: the launch date, Earth escape velocity, departure ascension and declination angles, burn index, and the time of flight for the first leg. Each gravity assist is then defined by four variables: the time of flight for each leg, the burn index, the B-plane insertion angle, and the radius ratio. A list of each variable in the integer and real-valued parts of each chromosome is given in Table 2.

Varying the number of flybys would typically be accompanied by a change in the length of both the integer and real-valued portions of the chromosome. However, genetic algorithms and NLP solvers are not designed to deal with variable length chromosomes. To overcome this problem, the hidden gene approach [17] is used. In this approach, some variables will not have an effect on the cost function for an individual solution. These hidden genes are still operated on by the genetic crossover and mutation operations for future generations. By using this hidden gene approach, the hybrid GA-NLP solver, presented in the following section, is able to optimize the number of gravity assists and each gravity-assist planet, as well as the real-valued variables associated with each mission type.

III. Global Hybrid Optimization Algorithm

Given the basics for each mission type and the accompanying objective function, the next step is to develop an optimization algorithm capable of determining global minimum solutions for MGA and MGA-DSM problems. The motivation for the optimization algorithm discussed in this section is to develop an algorithm that is capable of solving complex problems with no prior knowledge of the solution structure or approximate solutions, as is required for typical gradient-based nonlinear programming optimization. To accomplish this, a hybrid algorithm has been developed that is able to combine the global optimization capabilities of evolutionary algorithms with the local optimization capabilities of traditional NLP problem solvers. It will be shown that the final algorithm is capable of determining an optimal (or at least very near optimal) solutions for both MGA and MGA-DSM trajectories. This type of problem is known to be one of the most difficult optimization problems in astrodynamics and has been an area of active research in the recent years [12,24,32–34]. The hybrid GA-NLP algorithm is able to find complete solutions, including the number of gravity assists and flyby order, for the two- and three-impulse classes of problems.

A. Genetic Algorithm

The heart of a genetic algorithm is the stochastic simulation of natural selection, reproduction, and mutations found in natural evolution. In these simulations, genetic operators are used to "evolve" an initial population, through genetic operators, to determine the best fitness design [35]. The purpose of this section is to discuss the basics of the genetic algorithm developed for the final hybrid optimization algorithm. The genetic algorithm is responsible for the global optimization capabilities of the final algorithm.

A genetic algorithm is a stochastic optimization method based on the principles of evolution. Genetic algorithms perform a probabilistic search by evolving a randomly chosen initial population [36–38]. The population is just a series of sets of variables that are evaluated by the objective function, which was developed in the previous section. The advantage of using evolutionary methods over traditional optimization methods is that no initial solution is necessary. This helps ensure, but does not guarantee, that solutions are not confined to a single locally optimal solution. Genetic algorithms also perform well in complex nonlinear and discontinuous design spaces. Despite all the advantages, evolutionary algorithms also have a downside. They almost always require a greater number of cost function evaluations than traditional gradient-based methods, increasing the computational requirements. Additionally, evolutionary algorithms do not make use of gradients, so there is no proof of convergence. It should also be noted that genetic algorithms were developed for bound-constrained minimization problems and may not always perform well for unconstrained minimization when very large bounds are used. Many shortcomings of evolutionary algorithms are alleviated in the final hybrid algorithm. By using a local NLP gradient-based solver, the hybrid algorithm requires significantly smaller populations and provides at least locally optimal solutions.

The basic genetic operators include selection, reproduction, crossover, mutation, and elitism. The algorithm developed for this study can use a number of different user-selected selection, reproduction, crossover, and mutation methods. In a real-valued genetic algorithm (as implemented for this study), each variable is represented by either its integer or real number value. Each real and integer variable is referred to as a gene. A complete set of variables, which defines a solution to the problem, forms a complete chromosome. A single chromosome then corresponds to one member of the entire population, which can contain hundreds to thousands of chromosomes. This real-valued approach has an advantage over the traditional binary representation of variables because the real-valued variables do not have a discrete resolution. This is especially important for problems that are sensitive to small changes in the variables. For the pure GA, individual input variables include upper and lower bounds, the size of the population, and various other parameters to control the flow and output of the final optimization routine.

The first step in the evolutionary process is to use uniform random numbers to generate the initial population. This is done in a way that ensures a completely random, uniformly distributed initial population is used to start the problem. Each individual chromosome is generated using the integer and real-valued user-supplied upper and lower bounds. The process is then repeated until the entire population has been filled.

From this point, each member of the population is assigned a fitness value via a user-supplied objective function. The genetic operators are then used to generate a new population from the initial parent population. These genetic operators are crucial to the

Table 2 Integer- and real-valued variables for the MGA-DSM problem

Integer	Real-valued variables
Number of gravity assists n	Departure variables: J_0, v_∞, α_0, β_0, ϵ_0, T_0
Gravity-assist planet: $P_1, \ldots, P_n$	Time of flight for each gravity-assist leg: $T_1, \ldots, T_n$
Final arrival planet P_f	Burn index for each gravity-assist leg: $\epsilon_1, \ldots, \epsilon_n$
	B-plane insertion angle for each gravity-assist leg: $\gamma_1, \ldots, \gamma_n$
	Radius ratio for each gravity-assist leg: $r_{p1}, \ldots, r_{\mathrm{pn}}$

performance of the genetic algorithms because they enable a more fit population to be evolved from the initial population. This process continues until the genetic algorithm exit condition occurs, at which point the final population is returned. Common stopping conditions include limiting the number of generations, monitoring when the best-fit solution stops changing, or monitoring when the average cost function value approaches the population minimum. For this study, the optimization algorithm is run out until the best solutions have not changed for at least 25 generations. The sequence of operations of the core operations of the genetic algorithm is illustrated in Fig. 2. One secondary operator, elitism, is also required for the genetic algorithm. This operator simply ensures that the best-fit solution(s) are not lost from generation to generation by directly inserting the best members of the parent population into the next generation. Additional details of the genetic algorithm can be found in [20,39].

B. Nonlinear Programming Solver

Evolutionary algorithms, particularly genetic algorithms, are well suited for global optimization. However, when genetic algorithms are used to optimize MGA and MGA-DSM problems, they often only find solutions near the global optimum. Alternatively, nonlinear programming solvers typically only converge to locally optimal solutions. By modifying the genetic algorithm to use an NLP problem solver to determine locally optimal solutions, a robust global optimization algorithm can be developed. This hybrid algorithm, known as the GA-NLP optimization algorithm, is able determine near-globally optimal solutions by combining the global convergence of the genetic algorithm with accuracy of the nonlinear programming algorithm. The proposed algorithm is able to efficiently solve complex problems by significantly reducing the population size and number of generations required to converge on near-globally optimal solutions.

The GA-NLP algorithm should only be used to optimize functions that are continuous and at least twice differentiable, at least in the neighborhood of the proposed solution. The NLP problem solver does not iterate on integer variables because this often introduces large discontinuities. The genetic algorithm is used exclusively to optimize integer variables. These properties make optimizing both high- and low-thrust mission design problems good candidates for the proposed GA-NLP optimization algorithm.

The NLP problem solver implemented in the hybrid algorithm is based on the UNCMIN optimization algorithm, originally written in FORTRAN 77 and introduced in [40,41]. This algorithm is a quasi-Newton algorithm based on steepest descent methods, which require only the objective function values. However, to improve convergence and runtimes, this algorithm does allow user-supplied gradient and Hessians. The NLP algorithm implemented in the hybrid algorithm is an updated version of the original UNCMIN algorithm written in Fortran 90. Additional information on this algorithm can be found in [40,41].

As implemented, the UNCMIN algorithm is an extremely modular system of algorithms capable of multiple step selection strategies and multiple derivative calculation techniques (for both first- and second-order derivates). First-order derivatives can be calculated analytically, or by forward or central finite difference. The second derivative Hessian matrix can be calculated analytically, with a Broyden–Fletcher–Goldfarb–Shanno (BFGS) approximation [42–45], or by finite difference. The BFGS method approximates the Hessian from gradient information and is often used in quasi-Newton methods.

C. Hybrid Algorithm Implementation

The final hybrid GA-NLP algorithm is able to achieve significant performance increases, in terms of algorithm efficiency, over the standard GA implementation. This is possible because each individual population member is a optimized with the NLP problem solver rather than simply evaluating the cost function and relying on the GA for the entire optimization process. Through this process, the GA-NLP algorithm is able to combine the global convergence properties of the genetic algorithm with the local solution accuracy of the nonlinear programming solver. A flowchart of the hybrid GA-NLP algorithm is shown in Fig. 3. This algorithm flow is similar to the genetic algorithm, except the cost function evaluation step has been replaced by the NLP optimization step. In this step, a driver for the NLP problem solver is called to set the NLP user inputs for the particular problem. The NLP problem solver then determines a locally optimal solution and outputs the results to the genetic algorithm. After this, the genetic operators create a new population and the process is continued until the algorithm exits and outputs the final results.

IV. MGA and MGA-DSM Mission Design Examples

Optimal trajectories are determined for two example missions to test the performance of the GA-NLP problem solver. An MGA mission to Jupiter will first be analyzed. The reference mission is NASA's Galileo mission, which used three gravity assists to reach Jupiter. The entire heliocentric trajectory from Earth departure up to the Jupiter orbit insertion burn is modeled. The second mission optimized is an MGA-DSM mission to Saturn. This mission is close to the Cassini mission, which performed a total of four flybys before arriving at Saturn. The first two were at Venus, whereas the second and third flybys were at Earth and Jupiter, respectively. The MGA-DSM model is necessary for this mission because sufficient trajectories cannot be found with the MGA model. The total required ΔV can be significantly reduced by using the large deep-space maneuver (s) calculated with the MGA-DSM model. This mission is an ideal

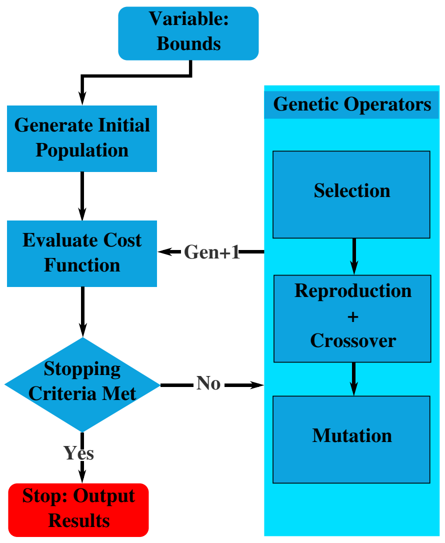


Fig. 2 Flowchart for the simple real- and integer-valued genetic algorithm.

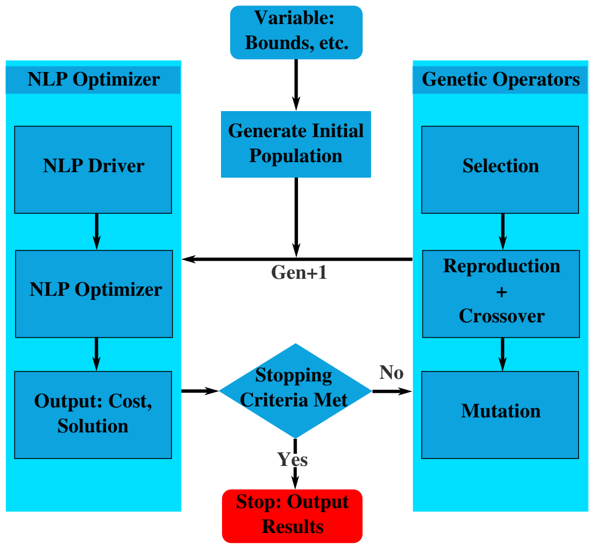


Fig. 3 Flowchart for hybrid GA-NLP algorithm.

candidate to test the performance of the GA-NLP optimizer because it is known to be one of the most difficult impulsive thrust missions to determine optimal trajectories [12,30,32].

A. Galileo Mission

Galileo was launched on 18 October 1989 on a mission to explore Jupiter. The spacecraft was launched from the Space Shuttle Atlantis using a interial upper stage. The combination of the spacecraft's high mass, which was over 2700 kg, and the Centaur upper stage placed tight constraint on the achievable Earth escape energy, or C_3 (which is simply v_∞^2). With the launch vehicle constraints, Galileo was forced to make use of interplanetary gravity assists to achieve sufficient energy to reach Jupiter. Three gravity assists were used: the first at Venus and the second two at Earth. The mission was further complicated by the use of a two-year resonant orbit between the two Earth gravity assists. The mission sequence flown by the Galileo spacecraft is often referred to as the Venus–Earth–Earth gravity-assist mission.

The Galileo mission is difficult to reproduce because of the two-year resonant orbit between the two Earth flybys. Solutions to Lambert's problem are unable to determine solutions when the orbits are colinear. Therefore, an alternate formulation to determine the flyby trajectories is used when the time of flight between two identical planets is within ± 2 days of a resonant orbit [31]. This allows the MGA algorithm to determine resonant orbits for any solutions during the optimization process. For this example mission, the GA-NLP optimization algorithm is able to determine an optimal trajectory with a two-year resonant orbit when the Earth–Venus–Earth–Earth–Jupiter (EVEEJ) flyby sequence was chosen by the algorithm approximately 100% of the time.

For this mission only, the ΔV required by the spacecraft was optimized. The Galileo mission had an upper C_3 limit of 17 $\mathrm{km}^2/\mathrm{s}^2$. Trajectories with a C_3 above 17 $\mathrm{km}^2/\mathrm{s}^2$ required an additional ΔV from the spacecraft. The objective function was also formulated to include the launch ΔV (only if the required C_3 was above 17 $\mathrm{km}^2/\mathrm{s}^2$), the ΔV required for each gravity assist, the final orbit Jupiter insertion ΔV, and any penalties associated with the flyby perigee radii and low-energy flybys. The arrival insertion burn was used to insert the spacecraft into a highly elliptical trajectory around Jupiter. The same Jupiter insertion orbit as the Galileo mission was used, with an eccentricity of 0.998 and a perigee radius of 286,000 km (the apogee was 10,776,000 km) [10].

To determine optimal trajectories for the Earth to Jupiter mission, the GA-NLP algorithm was used to determine the number of flybys, as well as the flyby order, launch date, and times of flight for each gravity-assist leg of the trajectory. To ensure that the algorithm only optimized an individual flyby order once, each optimal solution was saved. If the algorithm attempted the same flyby sequence again and the current solution had a cost function value greater than the saved trajectory optimal solution, the solution was thrown out. This simple check was performed by the objective function, so no modification of the general GA-NLP problem solver was necessary. If a solution was duplicated, and it was less optimal than the saved solution, the objective functions assigned a large value to the trajectory and the algorithm effectively ignored the solution by removing its genetic material from the population. This ensured that the algorithm determined a new flyby order each time the GA-NLP algorithm was called. The algorithm tested thousands of flyby orders during each run, but it typically converged on a particular flyby order with a few generations. This process was then repeated a few times, typically two to three, in order to reduce the chance that the algorithm converged on a local minimum for a given flyby sequence. In this way, a significant number of flyby orders could be analyzed during each run, but only one flyby order list was found by the end of the runs.

Through extensive testing, it was determined that a population size of 50 was sufficient for the GA-NLP algorithm to determine near-optimal trajectories for each flyby sequence. The algorithm was run until no significant change (10^{-5}) in the optimal solution had occurred for 25 generations. In a given flyby sequence, a solution would typically converge within 250 generations. The GA-NLP algorithm was run dozens of times to determine unique optimal trajectories.[‡] The total execution time for the entire search, when utilizing parallel processing on the CPU, was approximately 2 h on a standard Dell T3500 workstation with an Intel Xenon W3520 2.67 GHz processor.

Table 3 Problem bounds for Earth-to-Jupiter MGA mission

	N_{GA}	p_i	T_0	T_i	$\phi_{1,2}$
U_b	8	4	31 December 1992	1500	2π
L_b	2	1	1 January 1988	25	-2π

The variable limits for the launch date, number of flybys, flyby planets, and times of flight are shown in Table 3. For this mission, up to eight gravity assists were allowed, ranging from Mercury to Mars. A five-year launch window was searched, ranging from 1 January 1988 to 31 December 1992. Each time of flight for the flyby trajectory legs was allowed to range from 25 to 1500 days. Many of the missions found could only be considered Galileo-like missions because a total time-of-flight limit was not imposed. However, all of the top 10 trajectories found had a time of flight under 9 years. Trajectories with a large number of gravity assists had potential times of flight up to 32 years.

The top 10 mission architectures found during the search are shown in Table 4. In addition to finding a mission close to the Galileo Earth–Venus–Earth–Earth–Jupiter flyby sequence, two other missions were found with a required spacecraft ΔV under 600 m/s. The best sequence found had four flybys [Earth–Venus–Venus–Earth–Earth–Jupiter (EVVEEJ)] and a time of flight of only 4.7 years. However, the optimal launch date required a C_3 of approximately 17 $\mathrm{km}^2/\mathrm{s}^2$, which was near the maximum the Centaur upper stage could achieve with the Galileo spacecraft. The launch window for this mission scenario occurred in March of 1988, which was approximately 1.5 years earlier than the launch date of the actual mission. This mission required a ΔV that was approximately 8 m/s less than the actual Galileo flyby order, which was well with the margin of error for preliminary mission analysis. A third possible mission, which required a total ΔV of approximately 600 m/s, or about 40 m/s more than the calculated Galileo trajectory, was also found. This trajectory replaced the first Earth gravity assist with Venus to form an EVVEJ flight sequence.

The calculated trajectory is compared to the actual trajectory in Table 5. The results of the GA-NLP algorithm are extremely close to the actual mission shown, with two notable differences. The launch date found is approximately two and half weeks later than the actual mission, which is still within the 41-day launch window for Galileo [10]. Additionally, the final Jupiter arrival date is approximately 5 weeks early than the actual trajectory. Overall, the mission found by the GA-NLP algorithm is sufficient for preliminary mission analysis of the Galileo mission, with any differences likely due to differences in the patched conic and n-body gravity models. The final itinerary for the optimized Galileo trajectory is shown in Table 6.

This trajectory has a required launch where C_3 is approximately 13.5 $\mathrm{km}^2/\mathrm{s}^2$, which is the known optimal for the Galileo mission [10]. The total ΔV required by the spacecraft is approximately 560 m/s. As shown by the flight itinerary, the trajectory is ballistic after launch, requiring no large maneuvers before the Jupiter orbit insertion burn. The final trajectory would likely require a few small course trajectory correction maneuvers, but this would be part of the high-fidelity mission analysis performed after the initial trajectory has been determined. At the second Earth flyby, the spacecraft passes within approximately 300 km of the Earth. At first, this altitude appears alarming, but it is nearly the same flyby altitude of the actual trajectory. The other two flyby altitudes are also close to the altitudes of the actual trajectory. When the spacecraft arrives at Jupiter, an orbit insertion burn of 558 m/s is required. This is approximately 70 m/s

[‡]Evolutionary algorithms rarely converge on the optimal solutions with a single run, so multiple runs were used to increase the likelihood that the real optimal solution was found.

Table 4 Top flyby candidates for the Galileo Earth-to-Jupiter mission

Sequence	ΔV_{sc}, km/s	Launch date	TOF, years
EVVEEJ	0.550	22 March 1988	4.7
EVEEJ	0.558	4 November 1989	6.4
EVVEJ	0.599	12 May 1988	6.0
EMVEMJ	0.937	7 October 1992	6.5
EVEMJ	0.956	3 October 1989	4.9
EMEEMJ	1.085	9 October 1992	8.8
EMEMEEJ	1.222	10 April 1991	8.3
EVMEMVEJ	1.283	23 August 1991	4.5
EEMEMJ	1.300	5 July 1991	7.8
EEMEEJ	1.324	14 June 1991	8.1

Table 5 Comparison of the optimal Galileo mission with the actual Galileo mission

	Actual Galileo	GA-NLP optimal
Earth launch	18 October 1989	4 November 1989
Venus flyby	10 February 1990	21 February 1990
Earth flyby 1	8 December 1990	7 December 1990
Earth flyby 2	8 December 1992	7 December 1992
Jupiter arrival	8 March 1996	30 January 1996

Table 6 Calculated timeline of the optimal mission

Planet	Date	C_3, km^2/s^2	ΔV, km/s	Altitude. km
Earth	4 November 1989	13.54	— —	— —
Venus	21 February 1990	— —	0	18,004.4
Earth	7 December 1990	— —	0	4,711.2
Earth	7 December 1992	— —	0	297.8
Jupiter	30 January 1996	— —	0.558	— —

less than the 630 m/s burn performed by the Galileo spacecraft for the Jupiter orbit insertion. This trajectory model assumes impulsive burns, so the actual required ΔV will likely be larger than the calculated values because gravity losses and engine inefficiencies are not taken into account. The final trajectory for the optimal Galileo mission is shown in Fig. 4.

B. Cassini Mission

The second example is to reproduce the Earth to Saturn Cassini trajectory. This trajectory required at least one large deep-space maneuver, so the final mission had to be modeled with the MGA-DSM model. Cassini was launched on 15 October 1997 and performed four flybys [Earth–Venus–Venus–Earth–Jupiter–Saturn (EVVEJS)] before entering a highly elliptical Saturn orbit on 1 July 2004. Like Galileo, the Cassini mission required the use of gravity assists to gain the required energy to reach the outer planets. Without at least one deep-space maneuver, an affordable low-ΔV mission trajectory could not be determined. Therefore, determining a trajectory to Saturn that had a required spacecraft ΔV of approximately 1 km/s required the MGA-DSM model.

The final hybrid algorithm is able to determine optimal solutions for both the Galileo and Cassini missions. However, for the MGA-DSM problems, large populations (on the order of 25,000–50,000) are required to find the optimal solutions. This requires long runtimes, on the order of multiple days, because the NLP problem solver is called tens of thousands of times per generation. To improve the overall program efficiency, an alternate two-step formulation is used to determine optimal MGA and alternate MGA-DSM trajectories.

The idea behind the MGA-DSM model is that the total required ΔV can be reduced, when compared to standard MGA missions. Therefore, initial trajectory candidates can be determined from optimal MGA missions. A search for the MGA Earth to Saturn mission is first performed, using the same search algorithm as the Galileo MGA search (as before, only the actual required spacecraft ΔV is minimized). The results from this search are then used as the inputs to optimize the MGA-DSM missions with the GA-NLP algorithm. The launch dates and times of flight from the MGA search are used to determine the neighborhood in which the MGA-DSM search will be performed. The launch date is limited to ± 45 days of the optimal MGA launch date, and the times of flight are limited to $\pm 15\%$ of the optimal MGA mission. An outline of the basic search strategy is shown in the following. Using this search strategy, the total execution time for the entire parallel search algorithm is approximately 6 h on the same Dell T3500 workstation as before:

1) Simple GA: determine gravity-assist order
   a) Population: 50
   b) Maximum generations: 2500
   c) Variable bounds: completely open
2) GA-NLP: Determine final optimal solution
   a) Flyby order is given from the GA-NLP MGA search
   b) Solution neighborhood for the launch date and flight times is also given from the GA-NLP MGA search
   c) The additional MGA-DSM variable bounds are left completely open.
   d) Population: 50
   e) Maximum generations: 2500

The additional bounds for the MGA-DSM search are given in Table 7. The bounds for the initial MGA search are the same as those for the Galileo mission, except launch dates are limited from 1 January 1997 to 31 December 1999. Additionally, an upper C_3 of 18.1 km^2/s^2 is used, which is the C_3 of the actual Cassini mission. If the launch dates are not constrained close to the actual Cassini mission, the algorithm converges on a solution with an EVVEJS flyby sequence requiring a ΔV of approximately 750 m/s. Unfortunately, the launch date for this trajectory occurs approximately 2 years before the Cassini launch.

The final GA-NLP algorithm and search strategy outlined previously were able to reproduce the optimal Cassini trajectory. However, on approximately 50% of the runs, the initial MGA search converged on an alternate optimal flyby sequence. None of these alternate sequences produced MGA-DSM results better than the optimal Cassini trajectory. To overcome this, the top 10 MGA sequences were checked to ensure that all of the best flyby sequences were tested. A list of the optimal MGA Earth to Cassini trajectories candidates is shown in Table 8. This list is a compilation of five runs of the search algorithm. The optimal mission found for an EVVEJS flyby sequence had a total required ΔV of approximately 1.6 km/s, which was approximately 600 m/s more than the actual Cassini trajectory. Four solutions were found on some of the runs that have a required spacecraft ΔV lower than the optimal EVVEJS MGA mission. When these missions were analyzed using the MGA-DSM model with the GA-NLP problem solver, the best mission found was the EVVEJS flyby sequence, which is the same used in the actual Cassini mission.

A comparison of the final results with the actual Cassini trajectory is shown in Table 9. As this table shows, the two trajectories are nearly identical, with the optimal trajectory having a required ΔV slightly lower than the actual trajectory. Both trajectories have a large (approximately 450 m/s) deep-space maneuver during the Venus-to-

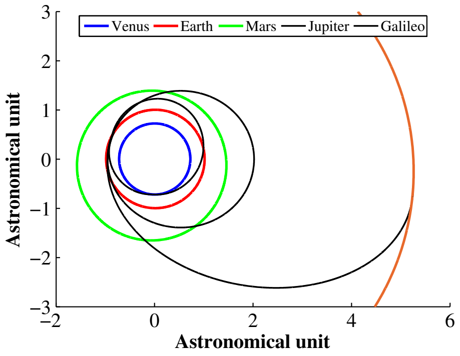


Fig. 4 Trajectory plot for the Galileo mission.

Table 7 Problem bounds for MGA-DSM missions

	C_{3L}	α, deg	β, deg	ϵ	γ, deg	r_{pi}
U_b	18.1	360	28.5	0.95	360	10,300
L_b	0	−360	−28.5	0.0001	−360	1.05

Table 8 Top 20 candidates for the Earth to Saturn MGA Missions

Sequence	ΔV_{sc}, km/s	Launch date	TOF, years
EVMVMVES	1.305	19 January 1998	10.6
EVVEVVES	1.307	8 February 1998	12.6
EVVES	1.411	19 February 1998	8.4
EVMEMVES	1.524	18 January 1998	11.6
EVVEJS	1.587	20 October 1997	7.3
EVEMES	1.619	1 March 1998	7.5
EMEVES	1.678	28 August 1997	9.0
EMVES	1.717	26 July 1999	9.1
EMEMVES	1.735	3 August 1997	11.0
EEMVES	1.764	27 July 1997	11.0

Table 9 Results for the Cassini mission compared to the actual results

Planet	Actual Cassini	GA-NLP optimal
Launch Date	6 October 1997	8 October 1997
C_3, km^2/s^2	18.1	18.0
DSM1, km/s	— —	0
Venus flyby	21 April 1998	23 April 1998
DSM2, km/s	0.466	0.444
Venus flyby 2	20 June 1999	21 June 1999
DSM3, km/s	— —	0
Earth	16 August 1999	17 August 1999
DSM4, km/s	— —	0
Jupiter	30 December 2000	6 January 2001
DSM5, km/s	— —	0
Saturn	1 July 2004	2 August 2004
Insertion burn, km/s	0.613	0.607
Total ΔV, km/s	1.079	1.051

Venus trajectory. The Saturn insertion ΔV for the actual mission is within 6 m/s of the optimal solution, which is well within the acceptable margins of error for preliminary mission analysis. However, the final arrival date for the optimal trajectory is approximately 1 month later than the actual mission. This is likely because the actual mission targeted an initial tour of Saturn's moons, which is not part of this analysis. The final trajectory is shown in Figs. 5a and 5b. Another likely contribution for this difference is that Saturn's sphere of influence is very large because of its large mass and distance from the sun.

Table 10 Performance of GA-NLP algorithm for Galileo and Cassini Missions

	N_{runs}	GA convergence, %	GA-NLP convergence, %
Galileo	10	60	100
Cassini	10	N/A	80

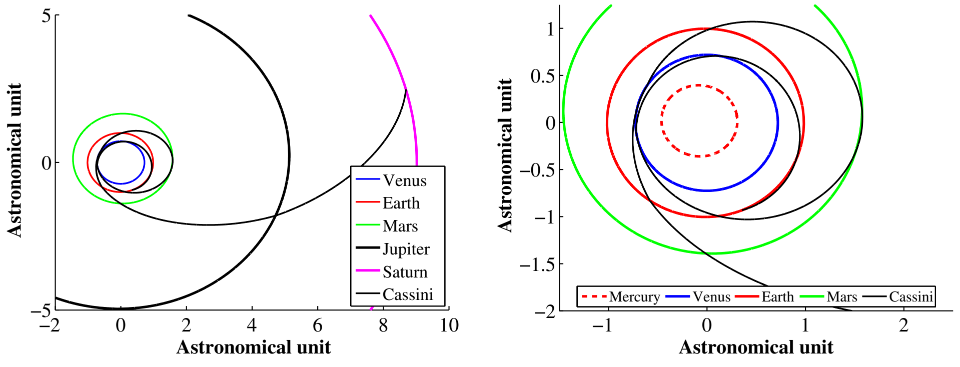
C. Performance of the GA-NLP Problem Solver

With optimal trajectories for both the Galileo and Cassini missions found by the GA-NLP algorithm, it is useful to examine the performance of the algorithm in determining the optimal MGA and MGA-DSM mission. For the Galileo mission, the algorithm was used to determine the top 20 mission candidates, including the number of flybys and all the decision variables for the mission. This test was then run up to 10 times, with the random number generated seeded with a different value each time, in order to determine how often the algorithm converges on the optimal solution. For each of the runs, the top five candidates from the GA-NLP algorithm were identical. The performances of both the standard genetic and GA-NLP algorithms for the Galileo mission are listed in Table 10. As this table shows, the standard genetic algorithm was able to determine the optimal Galileo mission 60% of the time, whereas the hybrid GA-NLP algorithm was able to converge on the optimal solution every time. It should be noted that, to determine the Galileo mission with the standard genetic algorithm, all constraints were added directly to the cost function. The standard genetic algorithm required significantly larger populations and more generations to converge: 1500 and 1000, respectively. Alternatively, the GA-NLP typically converged in 200–250 generations with a population size of only 50.

The GA-NLP algorithm was unable to converge on the optimal solution 100% of the time for the Cassini mission. With an 80% convergence rate, the algorithm should have been run several times to ensure that the optimal solution was found. However, the standard genetic algorithm was unable to converge on the optimal solution for this type of mission. With a population size of 100,000, that was allowed to evolve for 500 generations, the best solution the standard genetic algorithm was able to obtain was approximately 1.3 km/s [19]. Therefore, when compared to standard genetic algorithms, the GA-NLP algorithm was able to obtain significantly better performance for MGA-DSM missions, in terms of both determining optimal solutions and efficiency.

V. Conclusions

It has been shown that optimal preliminary interplanetary mission trajectories, with multiple possible gravity assists, can be determined with the proposed GA-NLP optimization algorithm. The GA-NLP algorithm combines a genetic algorithm with a nonlinear programming solver to robustly and efficiently determine near-optimal solutions to both MGA and MGA-DSM trajectories. In addition to the standard MGA and MGA-DSM models, a method to determine resonant orbits has also been implemented. This method allows the



a) Entire trajectory for the determined optimal mission **b) Trajectory of the inner-planetary fly bys**

Fig. 5 Trajectory for the optimal Earth-to-Saturn mission.

MGA model to be extended to include missions where resonant orbits result in lower ΔV missions. The hybrid algorithm is able to determine the number of gravity assists, as well as the flyby order, to determine optimal trajectories for each of potential mission. With this approach, thousands of possible trajectory combinations can be tested, ensuring that optimal flyby sequences are found. This GA-NLP-based search has been applied to both the Galileo and Cassini missions to produce near-optimal solutions for both test cases. For both missions, the trajectory found has a lower ΔV requirement than the actual mission flown.

This new GA-NLP hybrid-based search tool can be used for preliminary mission analysis of complex MGA and MGA-DSM missions. By using this optimization algorithm, complex missions can be analyzed on standard workstations without prior knowledge of the mission structure. This includes the number of flybys and possible flyby sequences. By using the GA-NLP algorithm, nonintuitive missions can be found that may be overlooked during typical preliminary mission studies.

Acknowledgment

This research has been supported in part by NASA's Iowa Space Grant Consortium.

References

[1] Hohmann, W., "The Attainability of Heavenly Bodies," NASA TT-F-44, Nov. 1960 (translated from German), https://archive.org/details/nasa_techdoc_19980230631.

[2] Goddard, R. H., "Three Astronautical Pioneers—1 Robert H. Goddard: An Autobiography," *Astronautics*, April 1959, pp. 24–27, 106–109, http://www.gravityassist.com/IAF3-1/Ref.%203-7.pdf.

[3] Tsiolkovskii, K. E., and Petroff, A. N., "Three Astronautical Pioneers—2 K. E. Tsiolkovskii: An Autobiography," *Astronautics*, May 1959, pp. 48–49, 63–64, http://www.gravityassist.com/IAF3-1/Ref.%203-6.pdf.

[4] Crocco, G. A., "One-Year Exploration-Trip Earth-Mars-Venus-Earth," *VII International Astronautical Congress*, Italian Assoc. of Aerotechnics/Italian Rocket Assoc., Sept. 1956, pp. 227–252.

[5] Ruppe, H. O., "Minimum Energy Requirements for Space Travel," *X International Astronautical Congress*, Springer–Verlag GmbH, New York, 1959, pp. 181–201.

[6] Battin, R., "Determination of Round-Trip Planetary Reconnaissance Trajectories," *Journal of the Aerospace Sciences*, Vol. 26, No. 9, Sept. 1959, pp. 545–567.
doi:10.2514/8.8204

[7] Lawden, D. F., "Interplanetary Rocket Trajectories," *Advances in Space Science*, Vol. 1, Elsevier, New York, 1959, pp. 1–53.
doi:10.1016/B978-1-4831-9959-7.50005-0

[8] Breakwell, J. V., Gillespie, R. W., and Ross, S., "Researches in Interplanetary Transfer," *American Rocket Society Journal*, Vol. 31, No. 2, 1959, pp. 201–208.
doi:10.2514/8.5428

[9] Minovitch, M., "The Invention that Opened the Solar System to Exploration," *Planetary and Space Science*, Vol. 58, No. 6, 2010, pp. 885–892.
doi:10.1016/j.pss.2010.01.008

[10] D'Amario, L., Bright, L., and Wolf, A., "Galileo Trajectory Design," *Space Science Review*, Vol. 60, Nos. 1–4, May 1992, pp. 23–78.
doi:10.1007/BF00216849

[11] Peralta, F., and Flanagan, S., "Cassini Interplanetary Trajectory Design," *Control Engineering Practice*, Vol. 3, No. 11, 1995, pp. 1603–1610.
doi:10.1016/0967-0661(95)00171-P

[12] Vasile, M., and De Pascale, P., "On the Preliminary Design of Multiple Gravity-Assist Trajectories," *Journal of Spacecraft and Rockets*, Vol. 43, No. 4, 2009, pp. 794–805.
doi:10.2514/1.17413

[13] Englander, J., Conway, B., and Williams, T., "Optimal Autonomous Mission Planning via Evolutionary Algorithms," *21th AAS/AIAA Space Flight Mechanics Meeting*, American Astronomical Soc. Paper 2011-159, Washington, D.C., Feb. 2011.

[14] Englander, J., Conway, B., and Williams, T., "Automated Mission Planning via Evolutionary Algorithms," *Journal of Guidance, Control, and Dynamics*, Vol. 35, No. 6, 2012, pp. 1878–1887.
doi:10.2514/1.54101

[15] Englander, J. A., Conway, B. A., and Williams, T., "Automated Interplanetary Mission Planning," *AAS/AIAA Astrodynamics Specialist Conference*, AIAA Paper 2012-4517, Aug. 2012.

[16] Vinkó, T., and Izzo, D., "Global Optimisation Heuristics and Test Problems for Preliminary Spacecraft Trajectory Design," Advanced Concepts Team, ESA TR ACT-TNT-MAD-GOHTPPSTD, Sept. 2008, http://www.esa.int/gsp/ACT/doc/INF/pub/ACT-TNT-INF-2008-GOHTPPSTD.pdf [retrieved 2015].

[17] Gad, A., and Abdelkhalik, O., "Hidden Genes Genetic Algorithm for Multi-Gravity-Assist Trajectories Optimization," *Journal of Spacecraft and Rockets*, Vol. 48, No. 4, July–Aug. 2011, pp. 629–641.
doi:10.2514/1.52642

[18] Abdelkhalik, O., and Gad, A., "Dynamic-Size Multi-Population Genetic Optimization for Multi-Gravity-Assist Trajectories," *Journal of Guidance, Control, and Dynamics*, Vol. 35, No. 2, March–April 2012, pp. 520–529.
doi:10.2514/1.54330

[19] Wagner, S., Kaplinger, B., and Wie, B., "GPU Accelerated Genetic Algorithm for Multiple Gravity-Assist and Impulsive V Maneuvers," *AIAA/AAS Guidance Navigation and Control Conference*, AIAA Paper 2012-4592, Aug. 2012.

[20] Wagner, S., and Wie, B., "Low-Thrust Trajectory Optimization for Asteroid Exploration, Redirct, and Deflection Mission," *24th AAS/AIAA Spaceflight Mechanics Meeting*, American Astronomical Soc. Paper 2014-283, Washington, D.C., Jan. 2014.

[21] Ceriotti, M., and Vasile, M., "MGA Trajectory Planning with an ACO-Inspired Algorithm," *Acta Astronautica*, Vol. 67, Nos. 9–10, 2010, pp. 1202–1217.
doi:10.1016/j.actaastro.2010.07.001

[22] Olympio, J. T., and Marmorat, J., "Global Trajectory Optimisation: Can We Prune the Solution Space When Considering Deep Space Maneuvers?" Advanced Concepts Team, ESA TR-06/4101, Sept. 2007.

[23] Vasile, M., and De Pascale, P., "Preliminary Design of Multiple Gravity-Assist Trajectories," *Journal of Spacecraft and Rockets*, Vol. 43, No. 4, 2006, pp. 794–805.
doi:10.2514/1.17413

[24] Qadir, K., "*Multi-Gravity Assist Design Tool for Interplanetary Trajectory Optimisation*, " M.S. Thesis, Lulea Univ. of Technology, Lulea, Sweden, 2009.

[25] Vallado, D., *Fundamentals of Astrodynamics and Applications*, 2nd ed., Microcosm Press, El Segundo, CA, 2004, pp. 448–470.

[26] Battin, R., *An Introduction to the Mathematics and Methods of Astrodynamics*, rev. ed., AIAA Educational Series, AIAA, Reston, VA, 1999, pp. 295–342.

[27] Lancaster, E. R., and Blanchard, R. C., "A Unified Form of Lambert's Theorem," NASA TN-D-5368, Sept. 1969.

[28] Sun, F. T., "On the Minimum Time Trajectory and Multiple Solutions of Lambert's Problem," *AAS/AIAA Astrodynamics Specialist Conference*, American Astronomical Soc. Paper 1979-164, Washington, D.C., June 1979.

[29] Gooding, R. H., "On the Solution of Lambert's Orbital Boundary-Value Problem," Royal Aerospace Establishment TR-88027, Farnborough, England, U.K., April 1988.

[30] Conway, B., *Spacecraft Trajectory Optimization*, 1st ed., Cambridge Univ. Press, New York, 2010, pp. 178–197.

[31] Davis, K., and Parker, J., "Constructing Resonant Orbits," Univ. of Colorado Boulder, ASEN 5519-Interplanetary Mission Design, Jan. 2010.

[32] Englander, J. A., "*Automated Trajectory Planning for Multiple Flyby Interplanetary Missions*, " Ph.D. Thesis, Univ. of Illinois, Urbana, IL, Aug. 2012.

[33] Petropoulos, A. E., Longuski, J. M., and Bonfiglio, E. P., "Trajectories to Jupiter via Gravity Assists from Venus, Earth, and Mars," *Journal of Spacecraft and Rockets*, Vol. 37, No. 6, 2000, pp. 776–783.
doi:10.2514/2.3650

[34] Zhu, K., Li, J., and Baoyin, H., "Multi-Swingby Optimization of Mission to Saturn Using Global Optimization Algorithms," *Acta Mechanica Sinica*, Vol. 25, No. 6, 2009, pp. 839–845.
doi:10.1007/s10409-009-0299-6

[35] Holland, J. H., *Adaptation in Natural and Artificial Systems*, Univ. of Michigan Press, Ann Arbor, MI, 1975, pp. 13–41.

[36] Goldberg, D., *Genetic Algorithms in Search, Optimization, and Machine Learning*, 1st ed., Addison Wesley, Reading, MA, 1989, pp. 59–88.

[37] Koza, J., *Genetic Programming: On the Programming of Computers by Means of Natural Selection*, 1st ed., MIT Press, Cambridge, MA, 1992, pp. 80–119.

[38] Syswerda, G., "Uniform Crossover in Genetic Algorithms," *Proceedings of the 3rd International Conference on Genetic Algorithms*, Morgan Kaufmann, San Francisco, 1989, pp. 2–9.

[39] Wagner, S., Winkler, T., and Wie, B., "Analysis and Selection of Optimal Targets for a Planetary Defense Technology Demonstration Mission," *AIAA/AAS Guidance Navigation and Control Conference*, AIAA Paper 2012-4874, Aug. 2012.

[40] Schnabel, R., Koontz, J., and Weiss, B., "A Modular System of Algorithms for Unconstrained Minimization," Dept. of Computer Science, Univ. of Colorado Boulder, Boulder, CO, 1985.

[41] Kahaner, D., Moler, C., and Nash, S., *Numerical Methods and Software*, 2nd ed., Prentice–Hall Series in Computational Mathematics, Prentice–Hall, Englewood Cliffs, NJ, 1989, Paper 07632.

[42] Broyden, C. G., "The Convergence of a Class of Double Rank Minimization Algorithms: 2. The New Algorithm," *Journal of the Institute of Mathematics and its Applications*, Vol. 6, No. 1, 1970, pp. 76–90.
doi:10.1093/imamat/6.1.76

[43] Fletcher, R., "A New Approach to Variable Metric Algorithms," *Computer Journal*, Vol. 13, No. 3, 1970, pp. 317–322.
doi:10.1093/comjnl/13.3.317

[44] Goldfarb, D., "A Family of Variable Metric Methods Derived by Variational Means," *Mathematics of Computation*, Vol. 24, No. 109, 1970, pp. 23–26.
doi:10.1090/S0025-5718-1970-0258249-6

[45] Shanno, D. F., "Conditioning of Quasi-Newton Methods for Function Minimization," *Mathematics of Computation*, Vol. 24, No. 111, 1970, pp. 647–650.
doi:10.1090/S0025-5718-1970-0274029-X